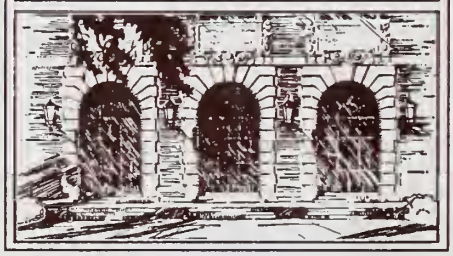


Report No. 372 COO-1469-0153

DEC PDP-8 INTERFACE TO IBM 2701 PDA

by

C. E. Carter
H. E. Lopeman
M. J. Michel
R. L. Miller

March 1970

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

Report No. 372

DEC PDP-8 INTERFACE TO IBM 2701 PDA*

by

C. E. Carter
H. E. Lopeman
M. J. Michel
R. L. Miller

March 1970

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

* This report supported in part by the Atomic Energy Commission under grant U. S. AEC AT(11-1)1469.

CONTENTS

PURPOSE

The purpose of this manual has two equally important parts. They are:

1. To serve as a maintenance manual for the DEC PDP-8 interface to IBM 2701-PDA hardware;
2. To be of some assistance to persons wishing to make the same processor connections.

0. INTRODUCTION

Soon after the installation of the DEC PDP-8/338 in 1967 it was clear that some communications link to a larger processor was needed. The first connection was made through a timesharing system into our "ILLIAC II" computer. The transfer rate for this connection was 100 characters/sec. in a half-duplex mode. All control and data was done through the accumulator with much overhead and delay.

When a change from "ILLIAC II" to 360 occurred plans to upgrade the PDP-8 communications were discussed. Since we had built an interface for a PDP-7 to a 360/50 through an IBM Controller (2701) we thought that the same type of interface to the PDP-8 might be useful. The basic IBM 2701 Controller can accommodate four separate interfaces of the type chosen so the PDP-8 and the PDP-7 could share the same IBM channel interface.

The hardware requirements for an interface such as this are imposed by the data break characteristics of a PDP-8 on one side and an IBM 2701 Parallel Data Adapter on the other. Since our particular PDP-8 is connected to many high speed I/O devices it was necessary to use a digital multiplexor as the interface hardware on the PDP-8 end of the connection. Since the two devices to be connected are

1. IBM 2701 Parallel Data Adapter, and
2. Digital Equipment Corporation Data Multiplexor DM01,

a short introduction to their signal connections will be given here.

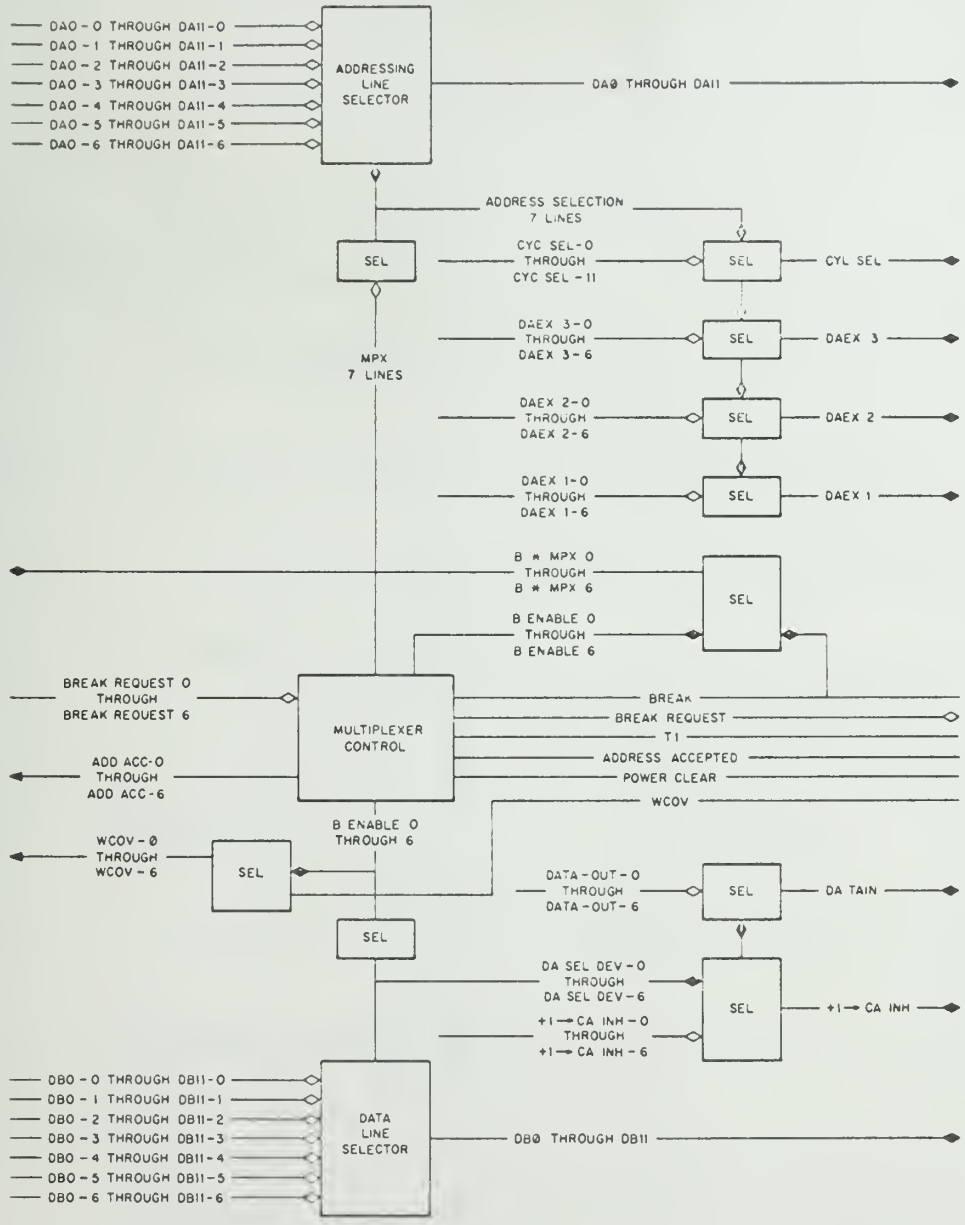
0.1 Multiplexor (DM01)

The (DM01) Data Multiplexor is a hardware switching system that connects to the data break facility of the PDP-8 and to one of seven peripheral devices. This unit can either be purchased from Digital Equipment Corporation or built by the customer. Its primary function is one of allowing rapid data-rate transfers to and from the core memory of the PDP-8 and some I/O device. Typical applications include fast disk, CRT refresh, magnetic tape drives, communications.

Two figures have been included here in an attempt to help make clear the position of the DM01. Figure 1 (Data Break Signals) shows the signals used by the data break and multiplexor. The top portion of the figure deals with address information. The bottom of the figure is for data while the center is for control. Figure 2 (Interconnecting Cable Diagram) shows the actual pin connections used on the connectors of the PDP-8 and DM01. The signal labeling for a typical device is shown as being in position "0" of the multiplexor. More detailed information along with the actual (DEC) circuit drawings can be obtained by requesting DEC-08-18AA-D.

0.2 Parallel Data Adapter

Operation between the Parallel Data Adapter and the external device is made through the Parallel Data Adapter interface. The interface consists of a set of lines which provide the control signals and data paths (Figure 3). The functions and pulse widths of each line of the interface are described below.



PERIPHERAL DEVICES

PDP-8

Figure 1. DM01 Data Break Signals

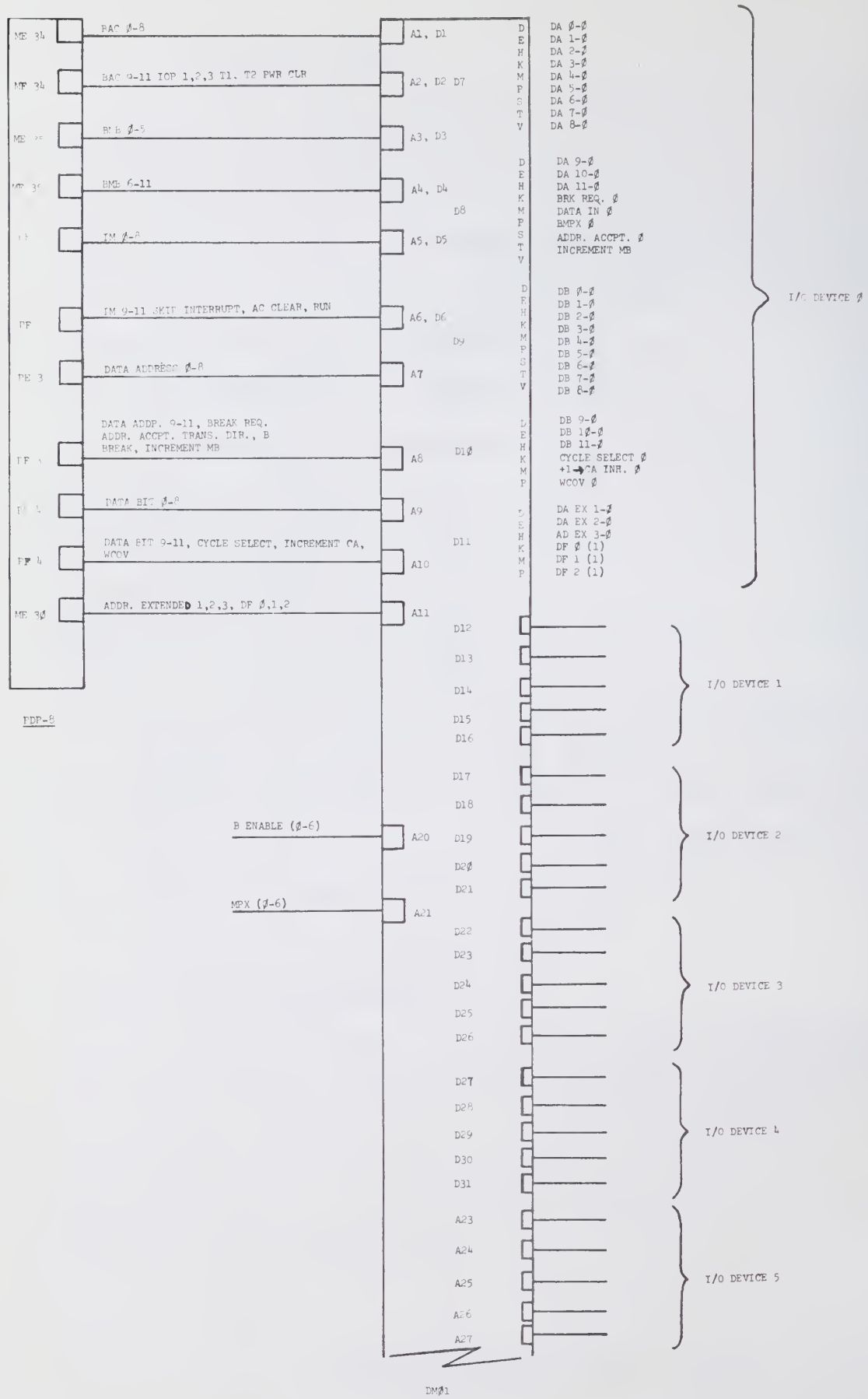

BAC Ø-8
BAC 9-11 IOP 1,2,3 T1. T2 PWR CLR
BMB Ø-5
BMB 6-11
IM Ø-8
IM 9-11 SKIP INTERRUPT, AC CLEAR, RUN
DATA ADDRESS Ø-8
DATA ADDR. 9-11, BREAK REQ. ADDR. ACCPT. TRANS. DIR., B BREAK, INCREMENT MB
DATA BIT Ø-8
DATA BIT 9-11, CYCLE SELECT, INCREMENT CA, WCOV
ADDR. EXTENDED 1,2,3, DF Ø,1,2
PDP-8
A1, D1
A2, D2 D7
A3, D3
A4, D4
D8
A5, D5
A6, D6
D9
A7
A8
D1Ø
A9
D11
A10
A11
DA Ø-Ø
DA 1-Ø
DA 2-Ø
DA 3-Ø
DA 4-Ø
DA 5-Ø
DA 6-Ø
DA 7-Ø
DA 8-Ø
DA 9-Ø
DA 10-Ø
DA 11-Ø
BRK REQ. Ø
DATA IN Ø
BMPX Ø
ADDR. ACCPT. Ø
INCREMENT MB
DB Ø-Ø
DB 1-Ø
DB 2-Ø
DB 3-Ø
DB 4-Ø
DB 5-Ø
DB 6-Ø
DB 7-Ø
DB 8-Ø
DB 9-Ø
DB 1Ø-Ø
DB 11-Ø
CYCLE SELECT Ø
+1→CA INH. Ø
WCOV Ø
DA EX 1-Ø
DA EX 2-Ø
AD EX 3-Ø
DF Ø (1)
DF 1 (1)
DF 2 (1)
I/O DEVICE Ø
B ENABLE (Ø-6)
MPX (Ø-6)
A20
A21
D12
D13
D14
D15
D16
D17
D18
D19
D2Ø
D21
D22
D23
D24
D25
D26
D27
D28
D29
D30
D31
A23
A24
A25
A26
A27
I/O DEVICE 1
I/O DEVICE 2
I/O DEVICE 3
I/O DEVICE 4
I/O DEVICE 5
DMØ1

Figure 2.

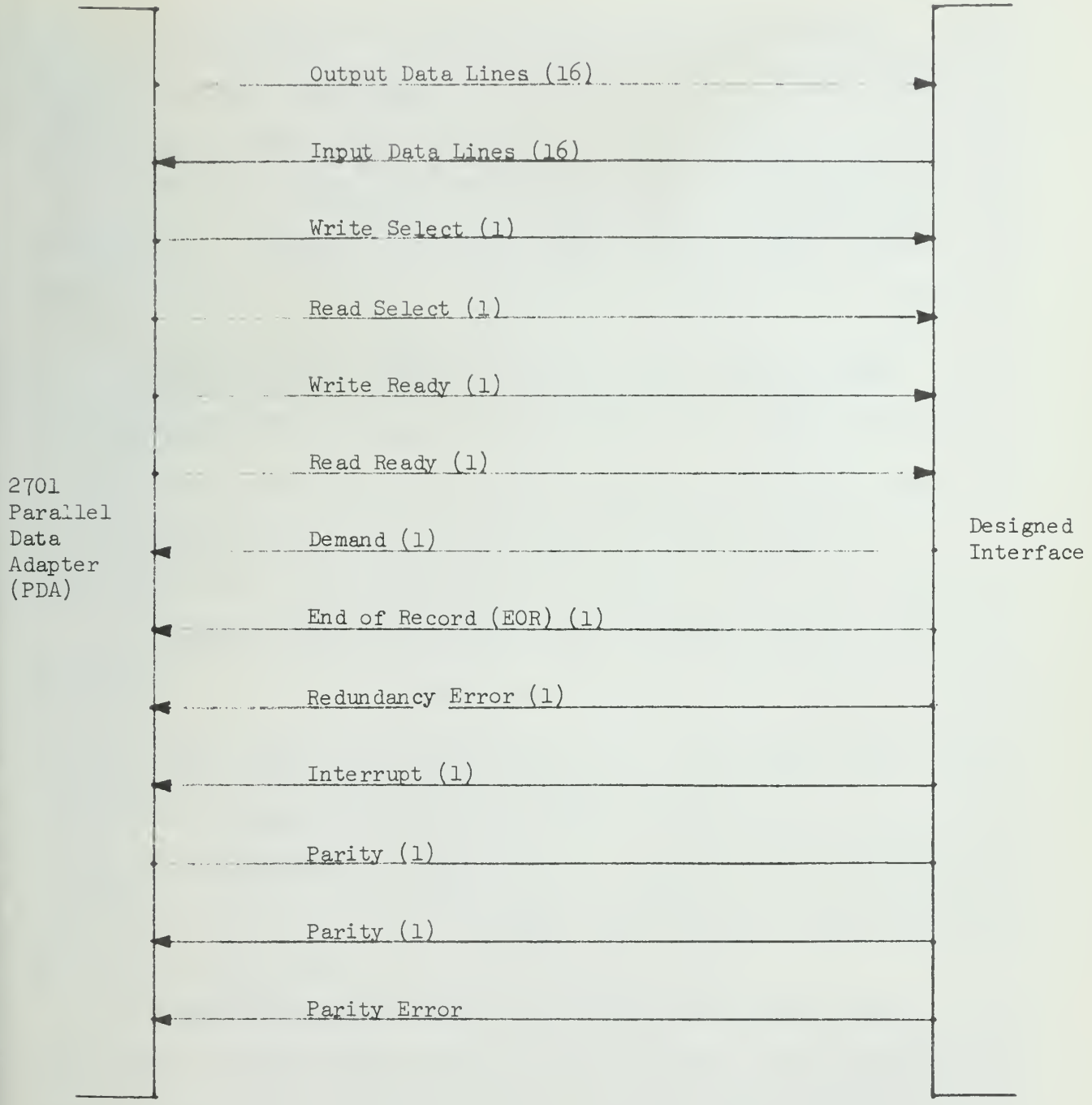


Figure 3. Parallel Data Interface

0.2.1 Output Data Bus (PDA to External Device): The output data bus consists of 17 lines on the basic adapter. Sixteen lines present the data word, and one line presents the odd parity bit to the external device. The output data bus is valid for sampling by the external device only when the Write Select and Write Ready lines are on. The bus will remain valid until the Demand signal is received.

0.2.2 Input Data Bus (External Device to PDA): The input data bus consists of 17 lines on the basic adapter. Sixteen lines are used for input data, and one line obtains the odd parity from the external device.

The Input Data Bus is sampled by the PDA when the Read Select and Demand lines are up. At this time, the signal lines on the data bus must be valid and deskewed at the 2701. The data bus must remain unchanged for the duration of the Demand signal.

0.2.3 Write Select (PDA to External Device): This line notifies the external device that it has been selected for a write operation. Recognition of a Write command from the channel causes the line to rise; it remains up until the end of the command.

0.2.4 Read Select (PDA to External Device): This line notifies the external device that it has been selected for a read operation. Recognition of the Read command from the channel causes the line to rise; it remains up until the end of the command.

0.2.5 Write Ready (PDA to External Device): This line notifies the external device that the data word is on the output data bus. The data is stabilized and deskewed before this line is raised.

The Write Ready line cannot come on while a Demand signal is still present. A Demand, EOR, or EOF signal from the external device resets this line.

0.2.6 Read Ready (PDA to External Device): This line notifies the external device that the PDA is ready to accept a word of data over the input bus.

Read Ready cannot come on while a Demand signal from the last data transfer under the same selection is still present.

A Demand, EOR, or EOF signal from the external device resets this line.

0.2.7 Demand (External Device to PDA): The significance of the Demand signal depends upon the command which is being executed.

Write Command: The Demand signal signifies that the external device has accepted the data word on the output data bus. There should be only one demand response for every Write Ready signal.

Read Command: The Demand signal signifies that the data on the input data bus is valid, stabilized, and deskewed. The data must remain valid for a minimum of 800 ns after the rise of the Demand signal.

Pulse Width:

Minimum: 800 ns

Maximum: The maximum limit is the desired data rate. The data rate can be no greater than the reciprocal

of the length of the demand pulse. This line must be down within 2 μs of the fall of the Read or Write Select line.

0.2.8 End of Record (EOR)(External Device to PDA): This line signifies that the external device has completed its operation and will not generate or accept any more data. Upon recognition, the PDA presents the Device End and Channel End status to the channel.

Pulse Width:

Minimum: 800 ns

Maximum: The EOR line must drop within 2 μs of the fall of Read or Write Select.

0.2.9 Redundancy Error (External Device to PDA): This line indicates that the external device has detected a parity error on a Write operation. This line may be up only when the Write Select line is up.

Pulse Width:

Minimum: 800 ns

Maximum: This line may stay up for the remainder of the operation, but must be reset within 2 μs of the fall of the Write Select line.

0.2.10 Interrupt (External Device to PDA): The Interrupt lines allow the external device to signal the CPU, through a Channel Interrupt, that it requires service.

1. HARDWARE

1.1 General

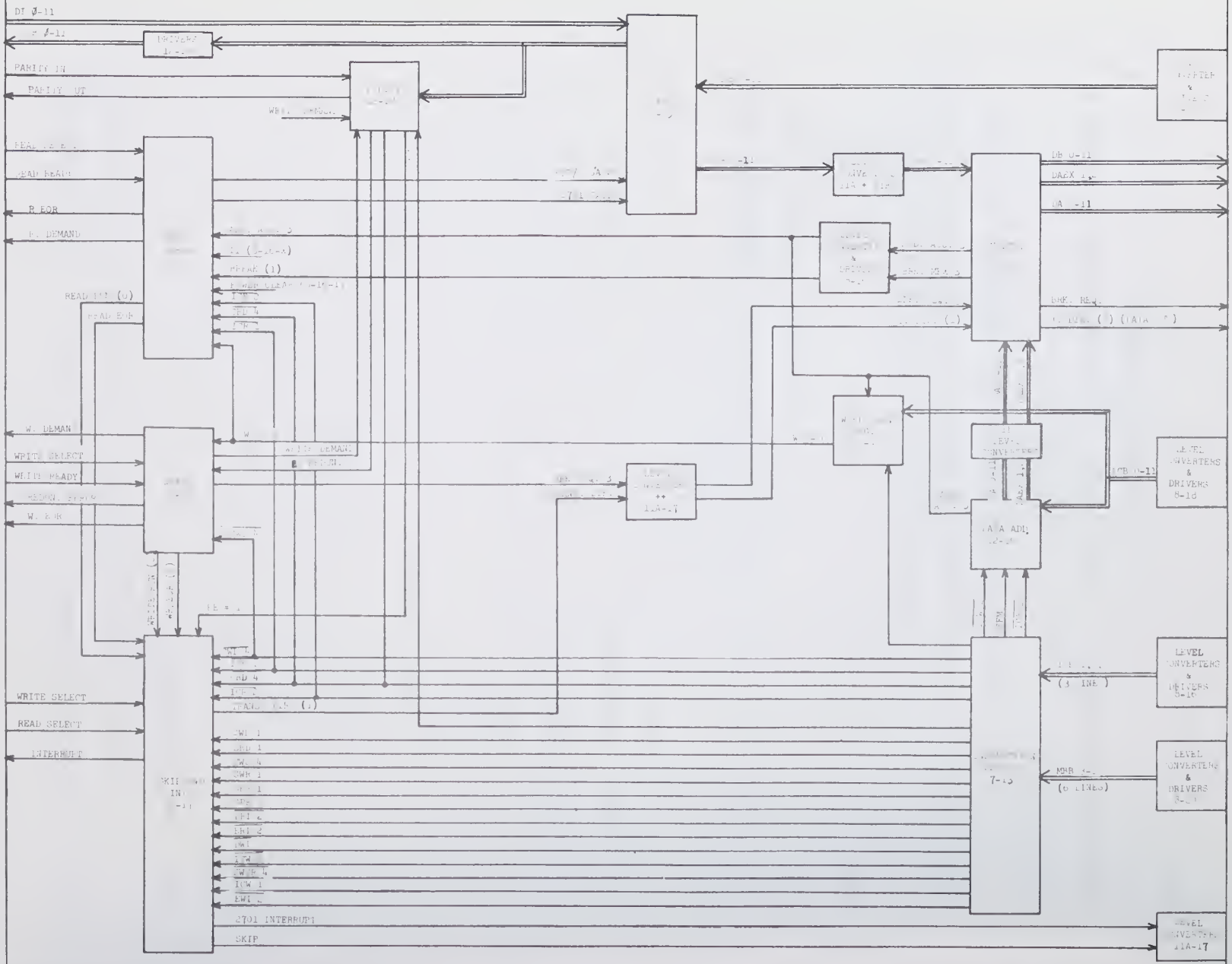
This section will be divided into two parts. A short discussion of the block diagram will be the first. The actual logic drawings will be the second.

Once it was decided that an IBM 2701 Parallel Data Adapter was to communicate with a Digital Equipment Corporation DM01, the general hardware requirements could be discussed. A rough block diagram was formed, the final version of which is Figure 4, to account for the various activities involved. From the block diagram it can be seen that, excluding the level converters, there are nine separate boxes or functional areas of the hardware interface. They are:

1. Instruction Decoder
2. Word Count
3. Data Buffer
4. Read
5. Write
6. Skip and Interrupt
7. Parity
8. Data Address
9. Multiplexor

1.2 Block Diagram

If there is some familiarity with the 2701 PDA on one end and the PDP-8 data break multiplexor on the other, there is little need for a



2701
PDA

PDP/8

BLOCK DIAGRAM

Figure 1

detailed explanation of this interface. The individual boxes of the block diagram refer to the specific logic drawing with numbers such as 12-18, 12-14, etc.

A statement or two will be made about each of the nine areas named above as an overview of the interface.

1.2.1 Instruction Decoder (7-13)

The instruction decoder logic is used to generate the various IOT signals and pulses for control and gating of the interface. The individual IOT's are shown on the instruction decoder drawing and explained in the section called "Programming of Channel."

1.2.2 Word Count Register (12-4)

The word count logic is used for signaling "WC=0" at the end of a block transfer. The block size is set from the accumulator of the PDP-8 by use of IOT (6424) "Set Word Count Register." The signal "address accepted" is used for the count to zero of the register. "WC=0" is used in Read logic and Write logic to indicate end of block.

1.2.3 Data Buffer (12-15)

The data buffer is a double gated 12 bit register used to buffer the data from the 2701 PDA and the PDP-8. Bits 0 through 11 from the PDP-8 memory buffer are gated into the register by "PDP-8 Gate." The output of the register is driven to the 2701 PDA by (12-14) drivers. The data from the 2701 PDA (DI bits) is gated into the register by the signal "2701 Gate." The output of the buffer register flops is tied into the multiplexor for entry to the PDP-8.

1.2.4 Read Logic (12-16)

The read logic is a part of a single card of logic containing Read and Write logic. The read portion of this card is the control logic for transfers to the 360 from the PDP-8. With the aid of drawing (12-16) the block diagram and the programming section of this report, the operation of the Read Control logic can be understood.

1.2.5 Write Logic (12-16)

The Write logic, sharing a single card with the Read logic, controls transfers from the 360 to the PDP-8. The operation of read and write are so similar that once one is understood the other will follow simply. The block diagram, the logic diagram (12-16) and the programming section of this report will aid in learning the Write operation.

1.2.6 Skip & Interrupt Logic (12-17)

The skip and interrupt logic is the area of the interface which utilizes most of the instruction decoder outputs. The logic primarily consists of several flip-flops and gates which are set or conditioned by instructions from the instruction decoder and levels from the 2701 PDA. These flops and gates in turn condition gates on the skip and interrupt busses so conditions may be checked by instructions from the PDP-8. The interrupt to the 2701 PDA is generated in this logic to inform the 2701 that the PDP-8 requires attention. Likewise, the PDP-8 interrupt is generated to notify the PDP-8 that the 2701 requires attention. The skip bus to the PDP-8 is activated when one of several conditions prevails and the PDP-8 does the proper instruction to check the condition.

The inputs from the instruction decoder are self-explanatory from their names. EWI (enable write interrupt) does just that, and ETC.

Read and Write EOR (end of record) come from the read control logic and write control logic, respectively. Their function is to set the read done and write done flops so the transfer end can be detected by the PDP-8. This is accomplished via the skip bus.

The read run(0) and write run(0) are conditioning levels for a choice of read or write interrupt to the PDP-8.

Read select and Write select from the 2701 PDA are used to generate an interrupt to the PDP-8 with proper conditions. These signals are also used to generate the transfer direction level to the multiplexor.

1.2.7 Parity (12-18)

The parity logic is used to calculate parity to send with the data word to the 2701 PDA. It is also used to calculate parity to compare with the parity sent by the 2701 PDA.

A parity error is looked at by the PDP-8 only during the write in from the 2701 PDA. The signal, Write Demand, is used as a gate. If the parity sent by the 2701 PDA and the calculated parity disagree, the PE (parity error) flop is set and the signal, Redundancy Error, is sent to the 2701 PDA. The parity error flop output is tied to the skip logic (12-17) and parity error check is normally made at the end of the transfer. The signal, CPE (clear parity error), is self-explanatory.

1.2.8 Data Address Register (12-20)

This logic is used to furnish a current address to the PDP-8 on a break cycle. The content of the PDP-8 accumulator (bits 0 through 11) is transferred to the address flops before a break cycle is initiated. If a write sequence is to be performed, the accumulator bits are gated into the address register by the signal ICW from the instruction decoder. The signal ICR gates the bits for a read sequence. The data address register is cleared to "0" by a delayed set. The address is incremented by the signal, Address Accepted, from the multiplexor logic.

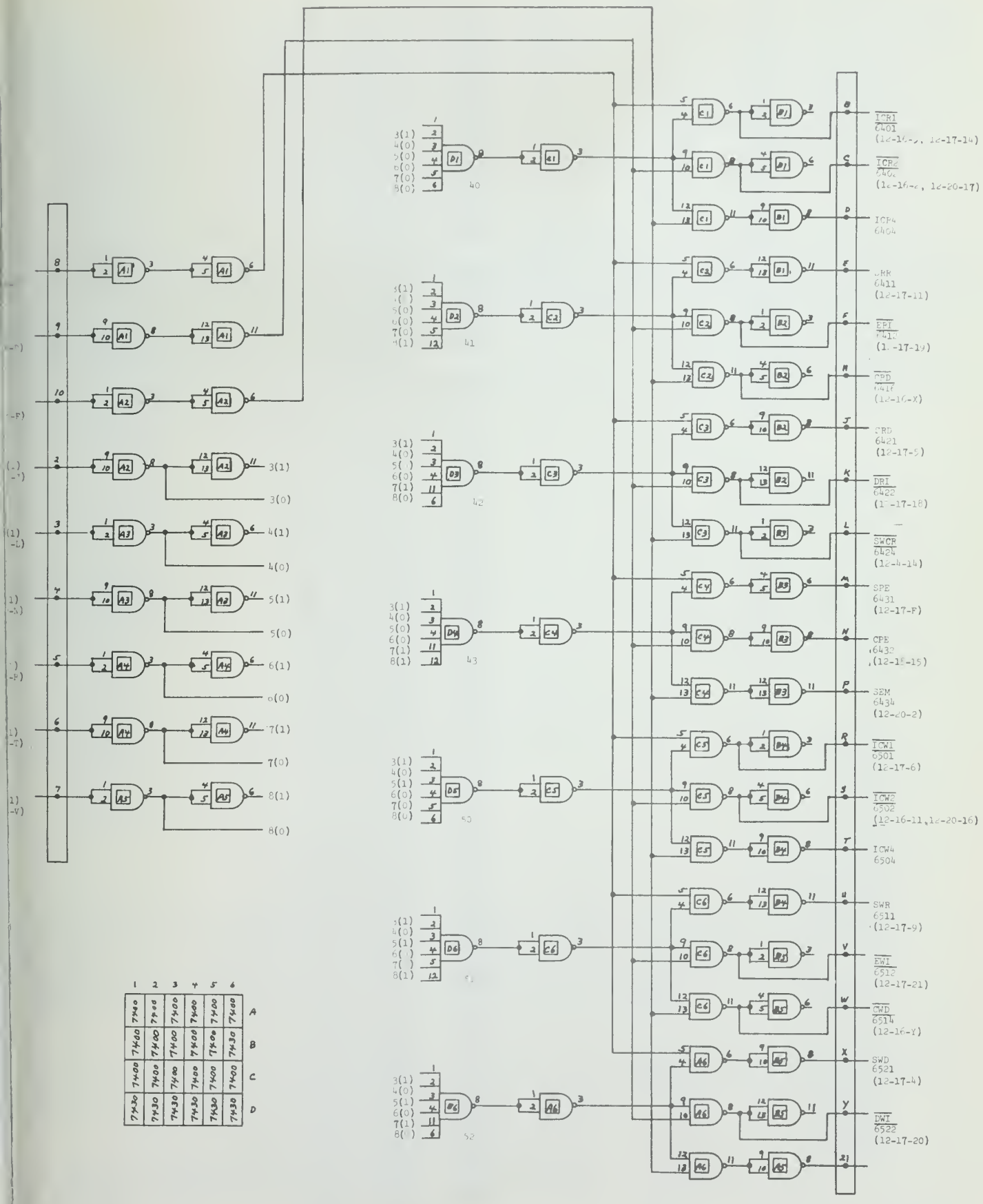
1.2.9 Multiplexor (DM01)

Because Digital multiplexors are reasonably simple and because more exact information can be received from DEC, little effort is made here to detail the multiplexor usage. Its position in the block diagram is rather clear.

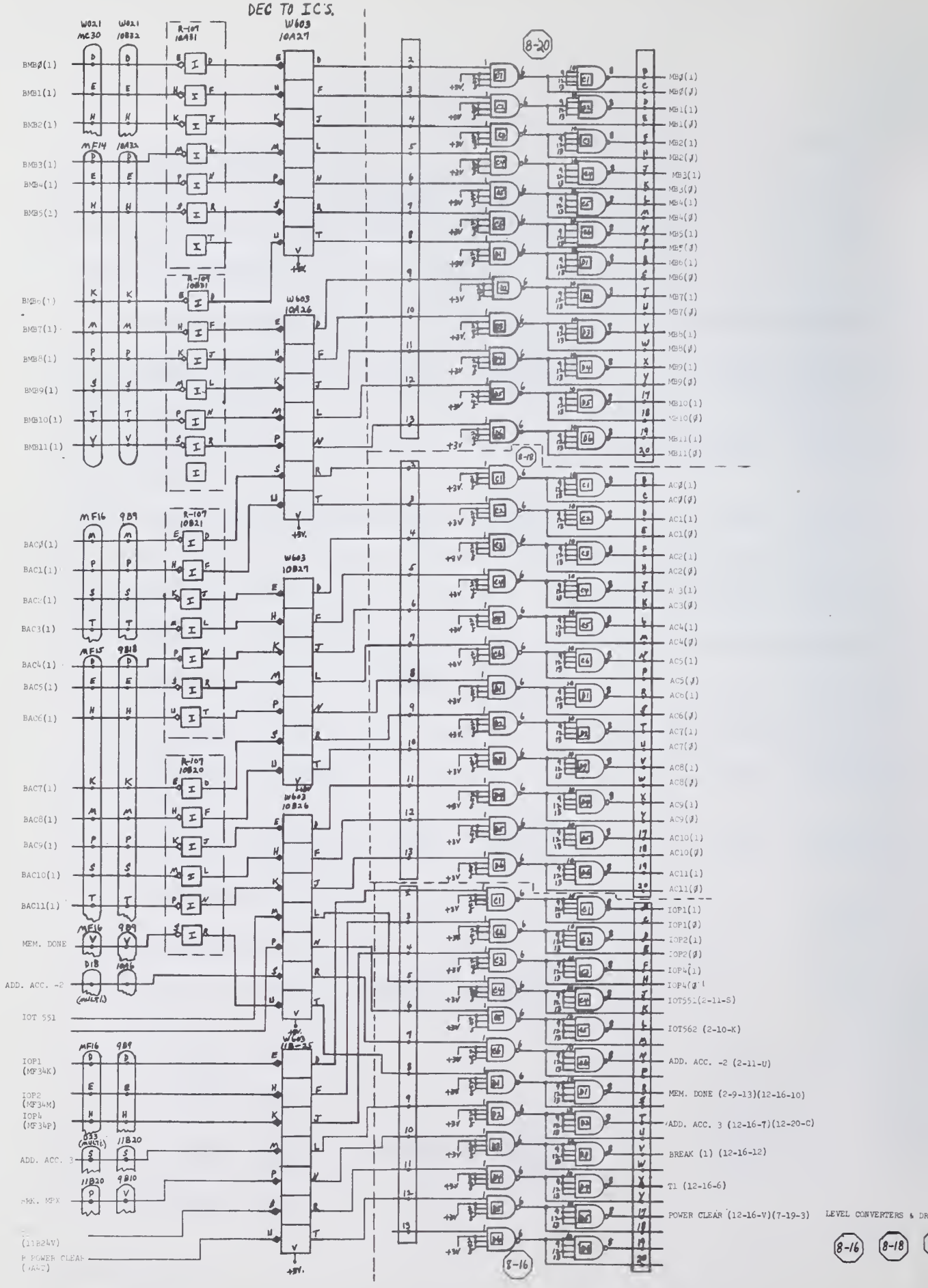
Copies of six of DEC's multiplexor drawings are included at the end of our logic drawings for the users' convenience.

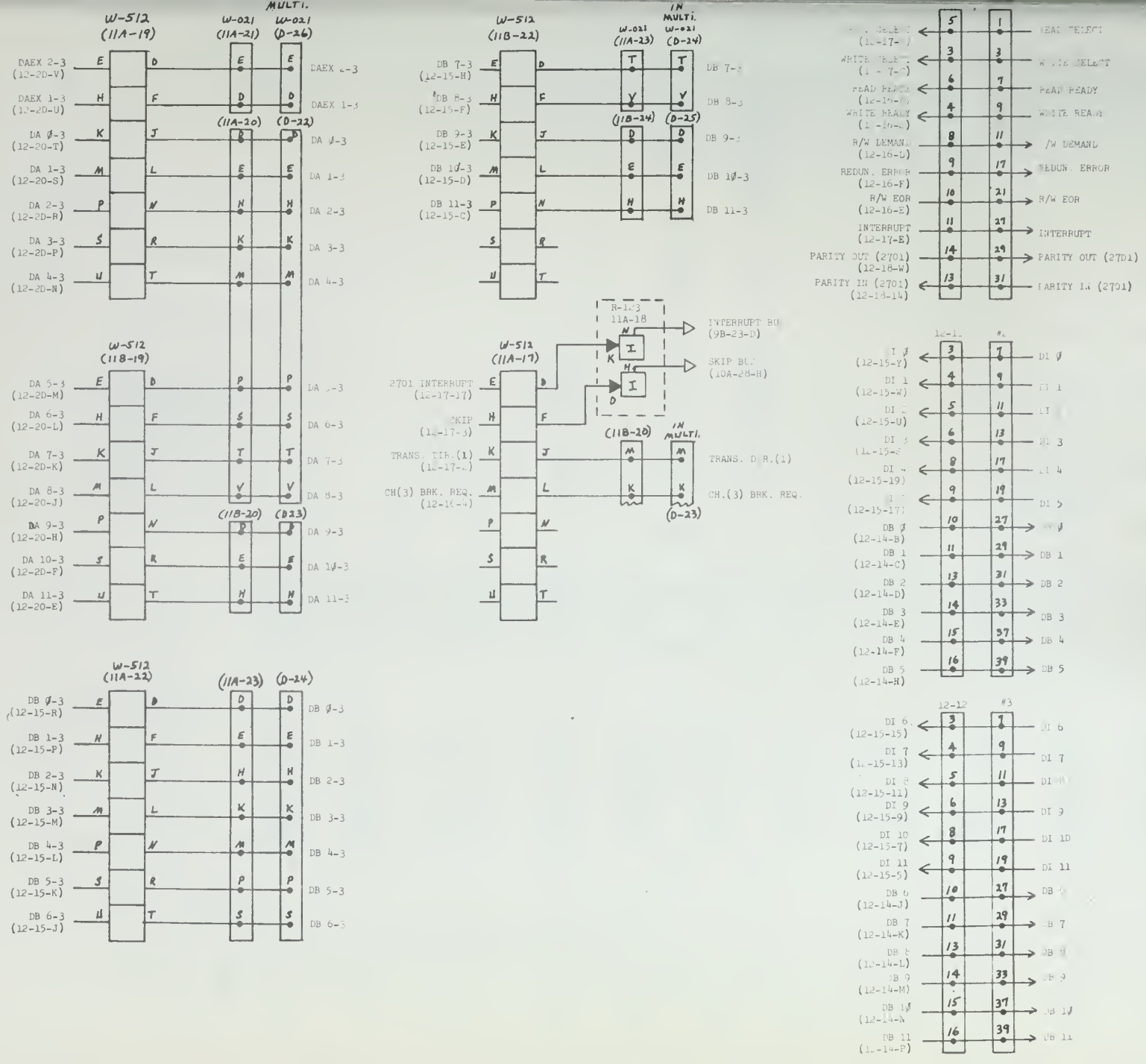
1.3 Logic Drawings

This section of hardware contains the individual logic drawings for the interface. Each drawing contains sufficient information such that by use of the block diagram the individual operations can be traced.

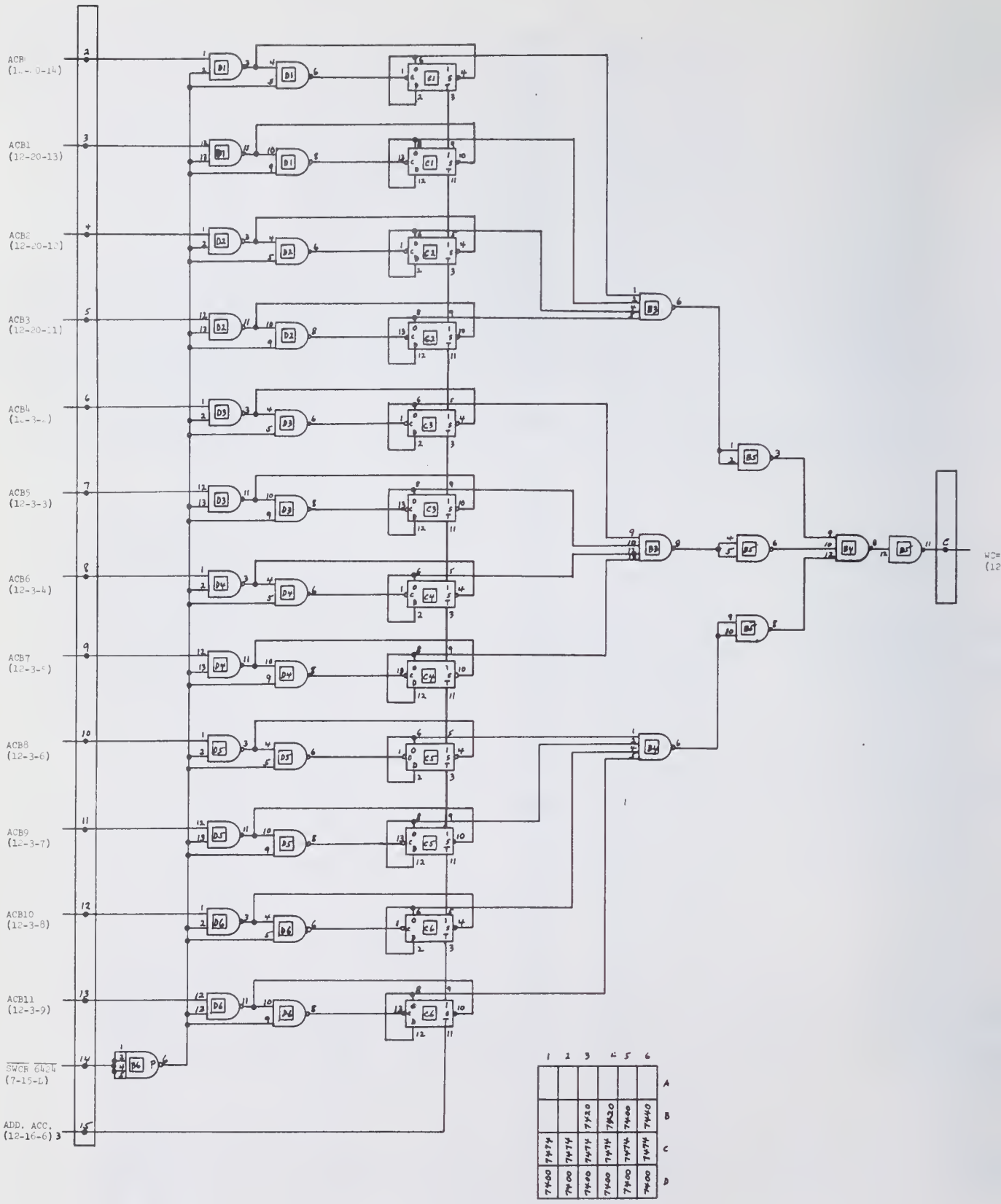


INSTRUCTION DECODER

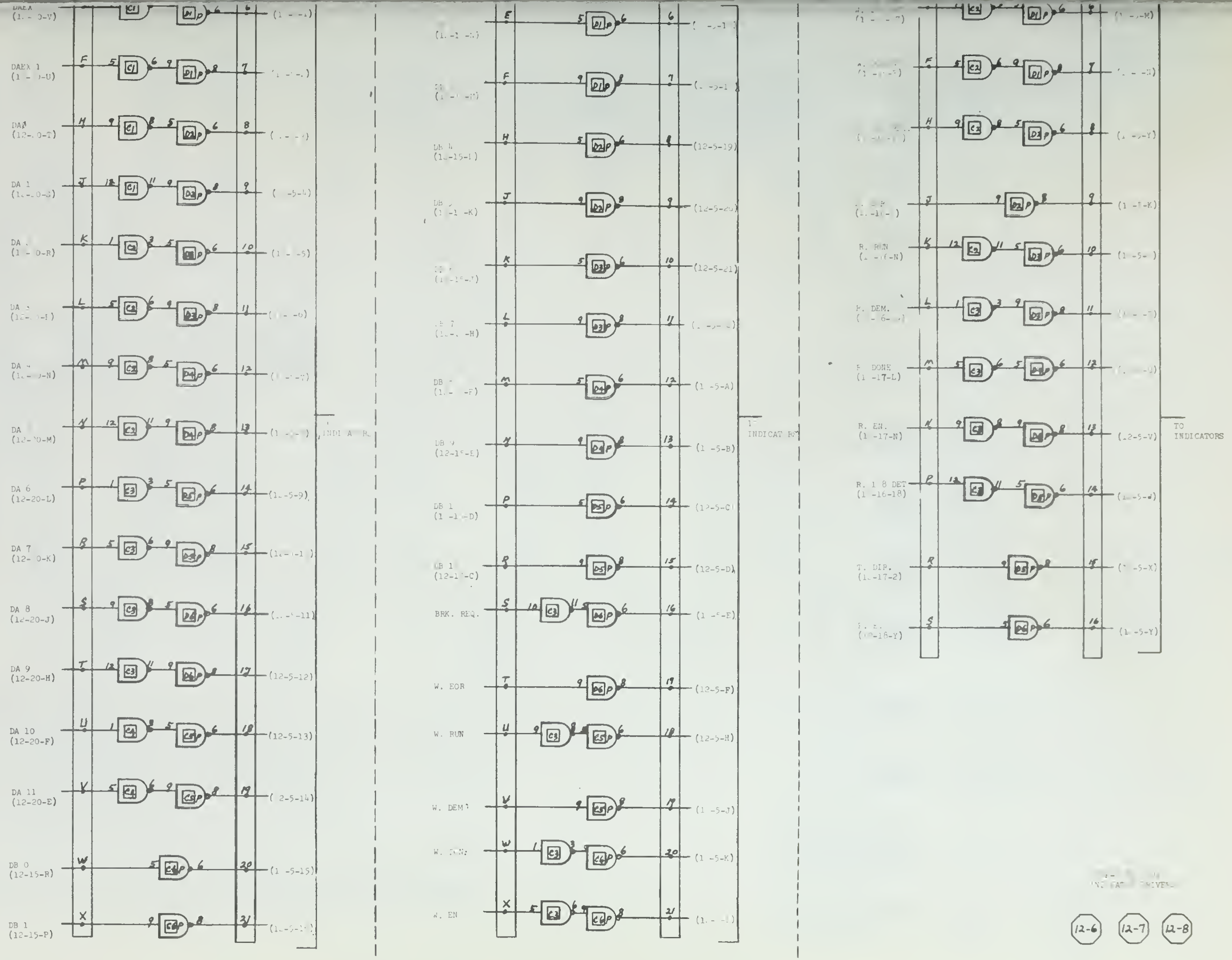

DEC TO IC'S.
W021 MC30
W021 10B32
R-107 10A31
W603 10A27
W603 10A26
W603 10B27
W603 10B26
W603 11B-25
MF14
MF16
MF15
9B9
9B8
9B10
11B20
DIB
(MULTI)
BMB0(1)
BMB1(1)
BMB2(1)
BMB3(1)
BMB4(1)
BMB5(1)
BMB6(1)
BMB7(1)
BMB8(1)
BMB9(1)
BMB10(1)
BMB11(1)
BAC0(1)
BAC1(1)
BAC2(1)
BAC3(1)
BAC4(1)
BAC5(1)
BAC6(1)
BAC7(1)
BAC8(1)
BAC9(1)
BAC10(1)
BAC11(1)
MEM. DONE
ADD. ACC. -2
IOT 551
IOP1 (MF34K)
IOP2 (MF34M)
IOP4 (MF34P)
ADD. ACC. 3
MEM. MPX
(11B24V)
POWER CLEAR (MA42)
+3V.
+8V.
8-20
8-18
8-16
MB0(1)
MB0(0)
MB1(1)
MB1(0)
MB2(1)
MB2(0)
MB3(1)
MB3(0)
MB4(1)
MB4(0)
MB5(1)
MB5(0)
MB6(1)
MB6(0)
MB7(1)
MB7(0)
MB8(1)
MB8(0)
MB9(1)
MB9(0)
MB10(1)
MB10(0)
MB11(1)
MB11(0)
AC0(1)
AC0(0)
AC1(1)
AC1(0)
AC2(1)
AC2(0)
AC3(1)
AC3(0)
AC4(1)
AC4(0)
AC5(1)
AC5(0)
AC6(1)
AC6(0)
AC7(1)
AC7(0)
AC8(1)
AC8(0)
AC9(1)
AC9(0)
AC10(1)
AC10(0)
AC11(1)
AC11(0)
IOP1(1)
IOP1(0)
IOP2(1)
IOP2(0)
IOP4(1)
IOP4(0)
IOT551 (2-11-S)
IOT562 (2-10-K)
ADD. ACC. -2 (2-11-U)
MEM. DONE (2-9-13)(12-16-10)
ADD. ACC. 3 (12-16-7)(12-20-C)
BREAK (1) (12-16-12)
T1 (12-16-6)
POWER CLEAR (12-16-V)(7-19-3)
LEVEL CONVERTERS & DR
8-16
8-18

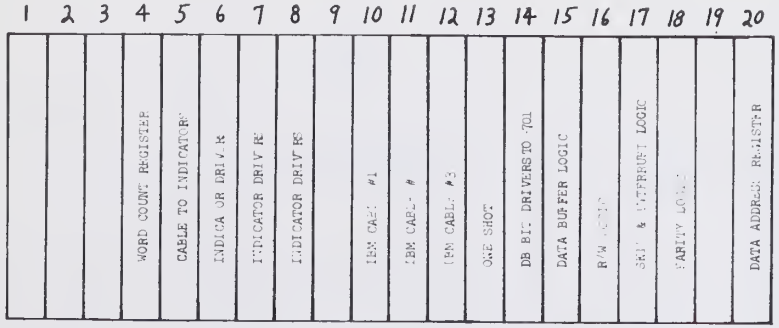


LEVEL CONVERTERS & CABLE

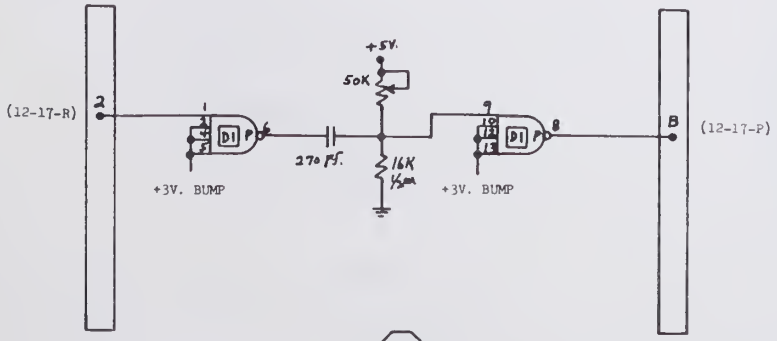


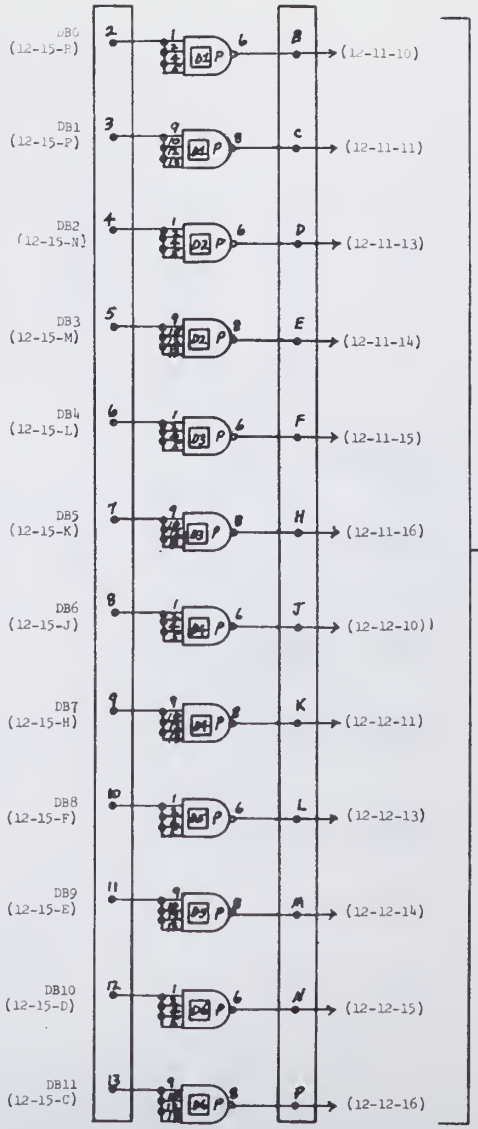
2701 - WORD COUNT REGISTER

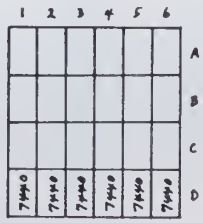

DAEX 1
DA 6
(12-20-L)
DA 7
DA 8
DA 9
(12-20-H)
DA 10
(12-20-F)
DA 11
(12-20-E)
DB 0
(12-15-R)
DB 1
(12-15-P)
(12-5-13)
(12-5-12)
(12-5-14)
BRK. REQ.
W. EOR
W. RUN
W. DEM
W. EN
(12-5-19)
(12-5-21)
(12-5-D)
(12-5-F)
INDICATORS
R. EN.
R. DEM.
R. RUN
T. DIP.
TO
INDICATORS
12-6
12-7
12-8



CARD LAYOUT



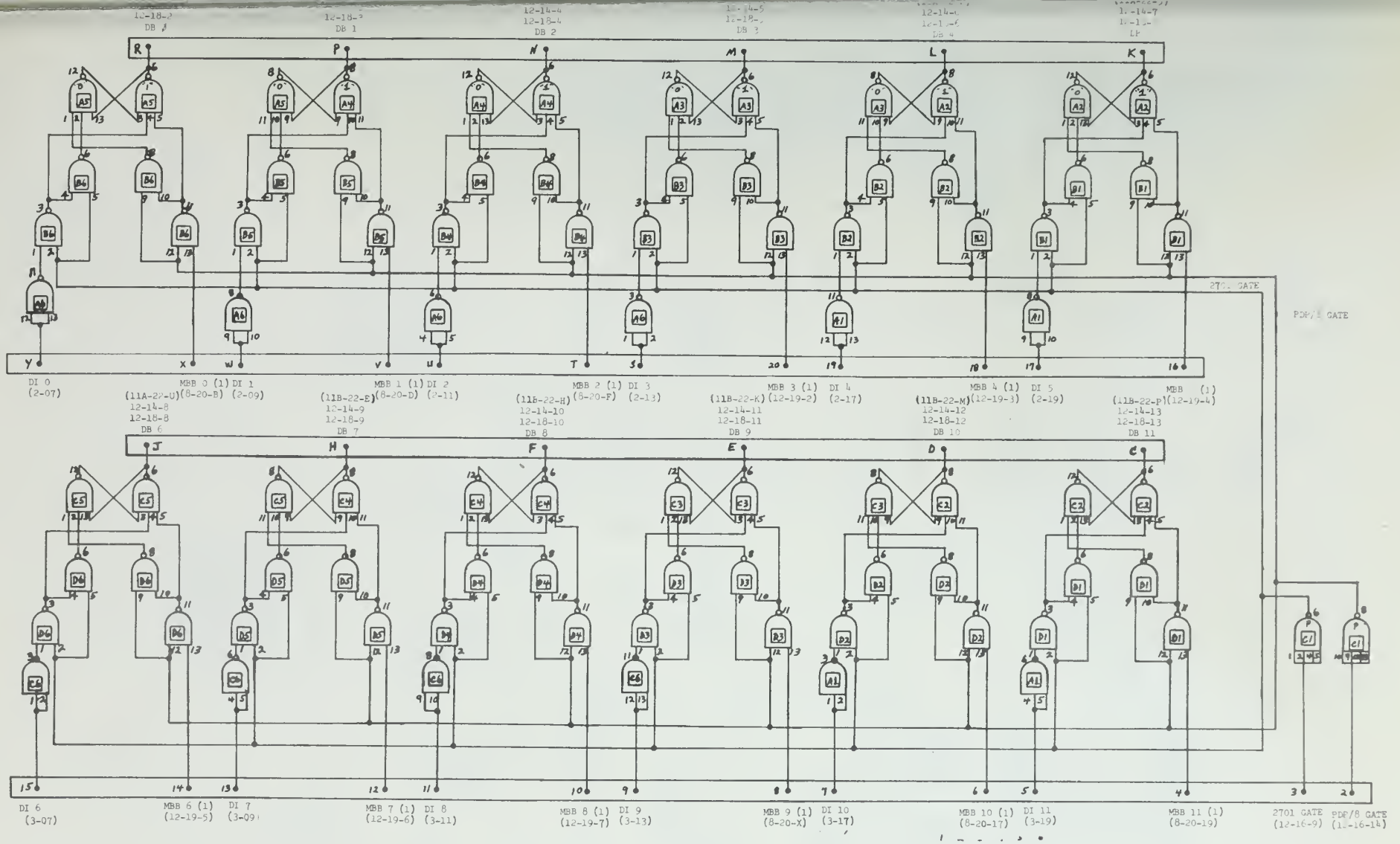




PDP/8 INTERFACE
TO 2701

DB BIT DRIVERS
TO 2701

12-13 12-14

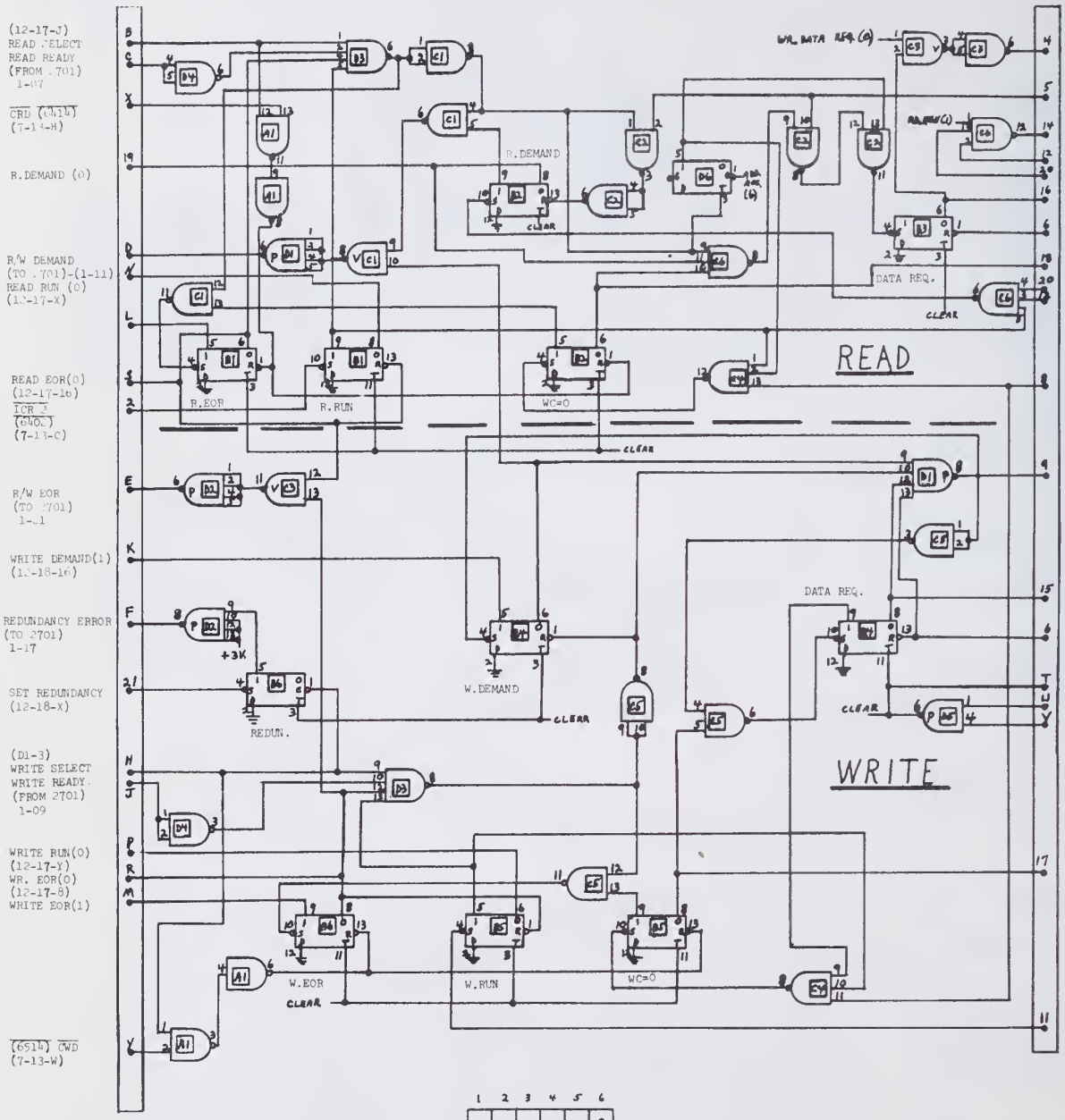


	1			2	
A	7400	7410	7410	7410	7400
B	7400	7400	7400	7400	7400
C	7400	7410	7410	7410	7400
D	7400	7400	7400	7400	7400

NOTE:

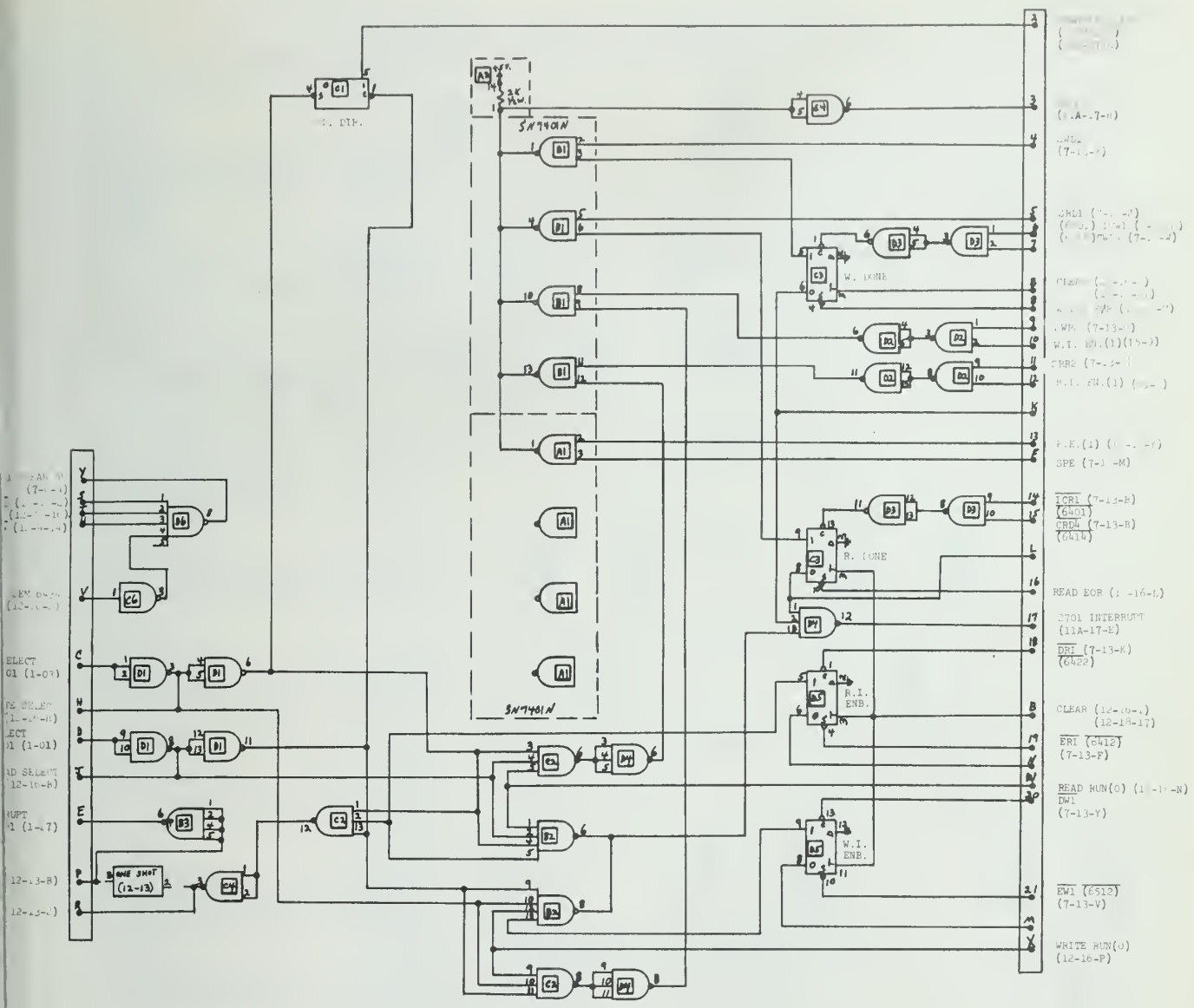
DI BITS ARE FROM IBM CABLE. FOR EXAMPLE: (3-07) CABLE NO. 3, PIN 7.

PDP-8 INTERFACE
DATA BUFFER



1	2	3	4	5	6	
					7440	A
7414	7414	7414	7474	7474	7474	B
7410	7400	7410	7400	7400	7400	C
7474	7440	7400	7430	7400	7440	D

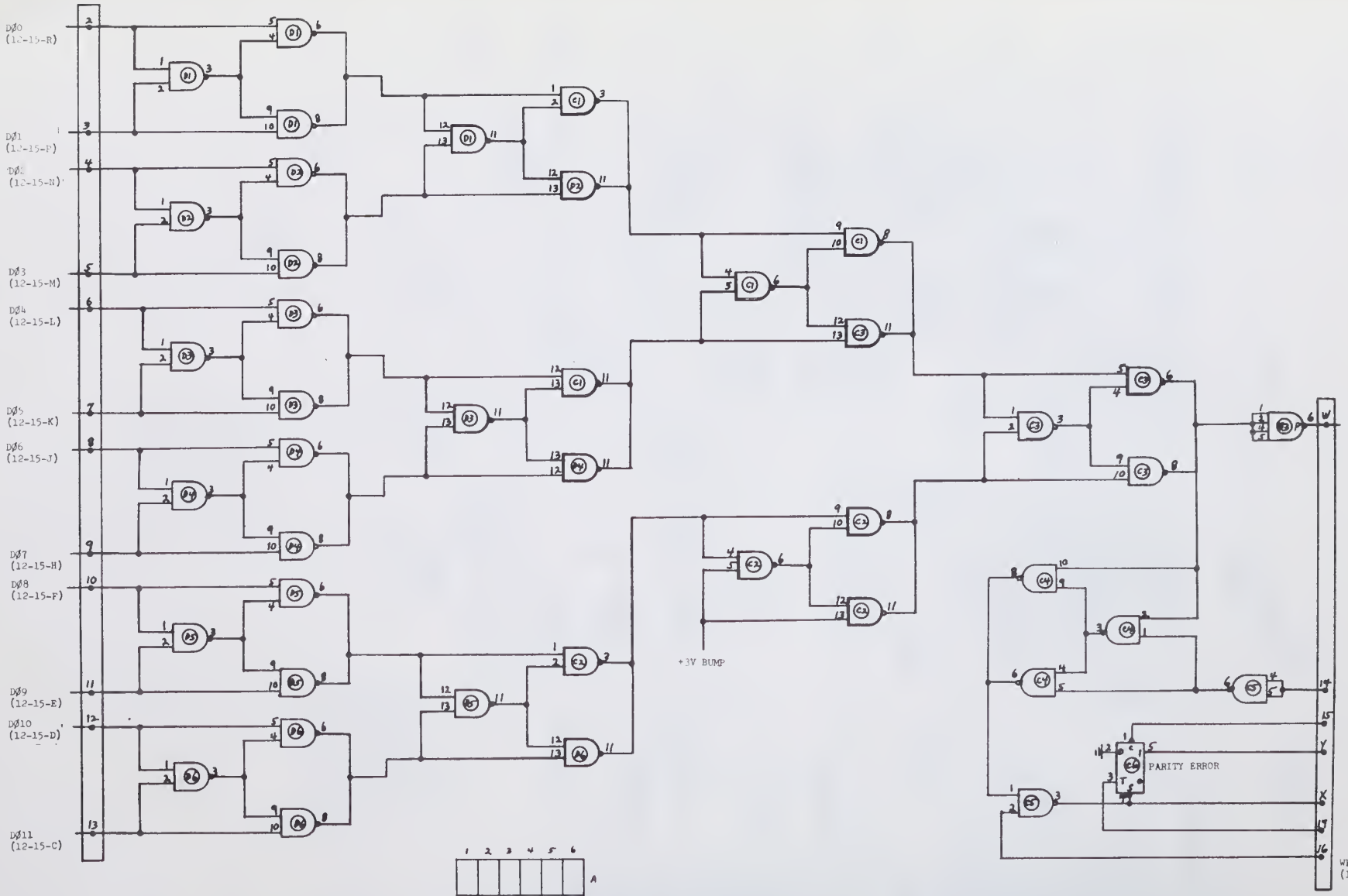
PDP/8 INTERFACE TO 2701
R/W LOGIC



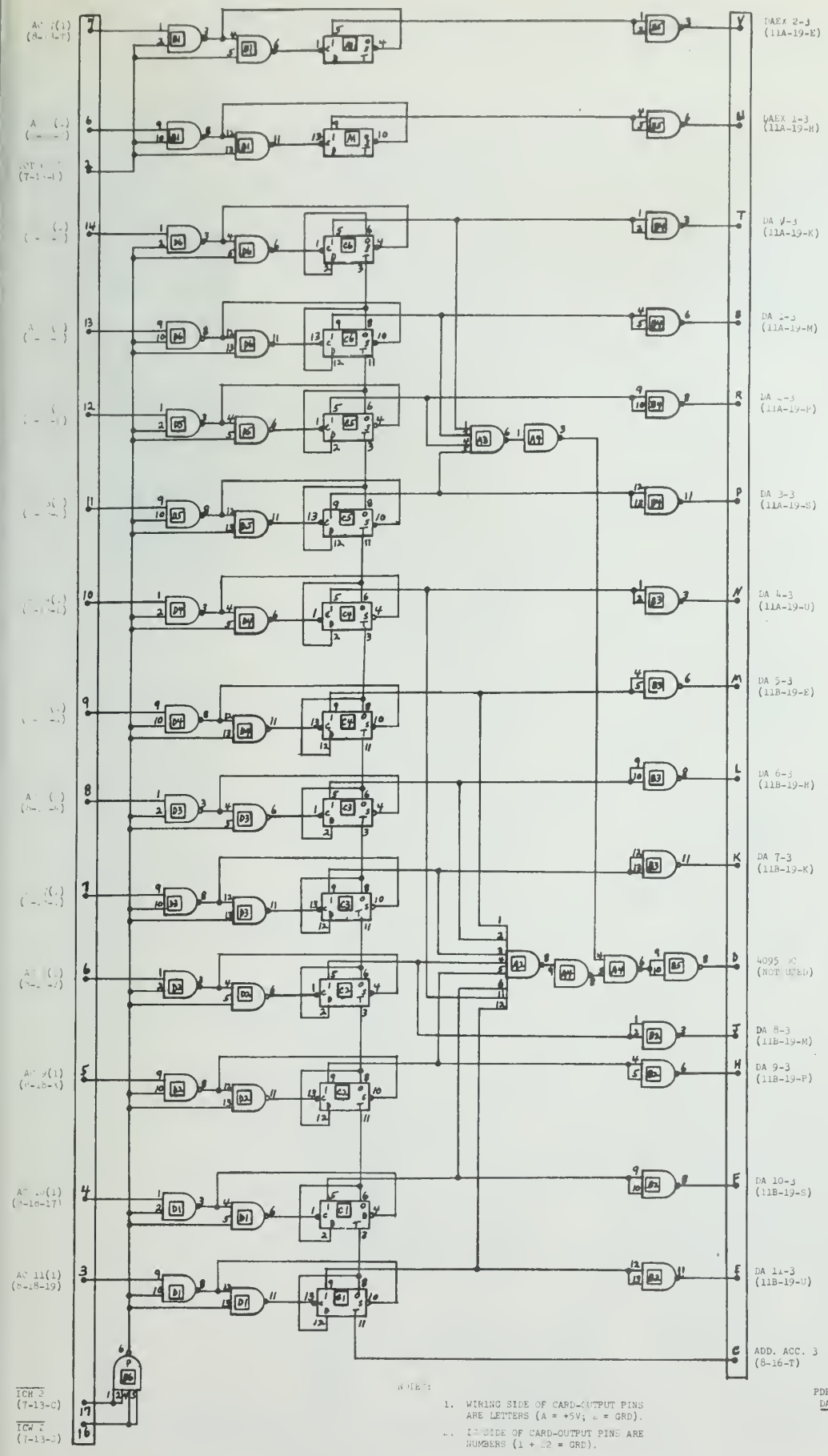
1	2	3	4	5	6	
7401	2K RESIST					A
7401	7402	7404				B
7447	7410	7447	7400		7400	C
7400	7400	7447	7401	7447	7854	D

PDP-8 INTERFACE TO 2701
SKIP & INTERRUPT LOGIC

12-19



PDP/8 INTERFACE TO 2701
PARITY LOGIC

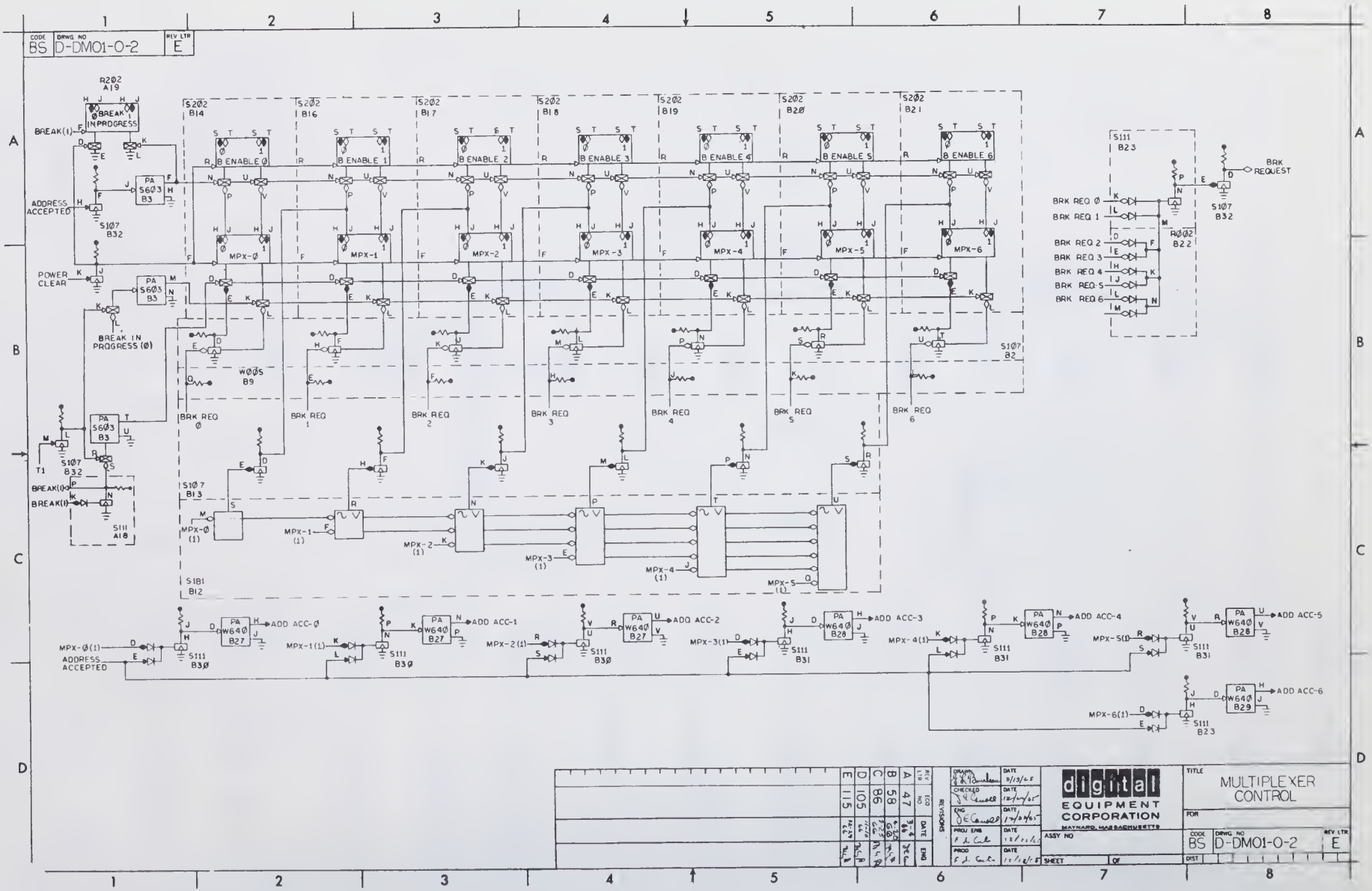


1	2	3	4	5	6	
7474	03E	7410	7400			A
0000	7404	7400	7410	7400	7410	B
7474	7474	7474	7474	7474	7474	C
7400	7400	7400	7400	7400	7400	D

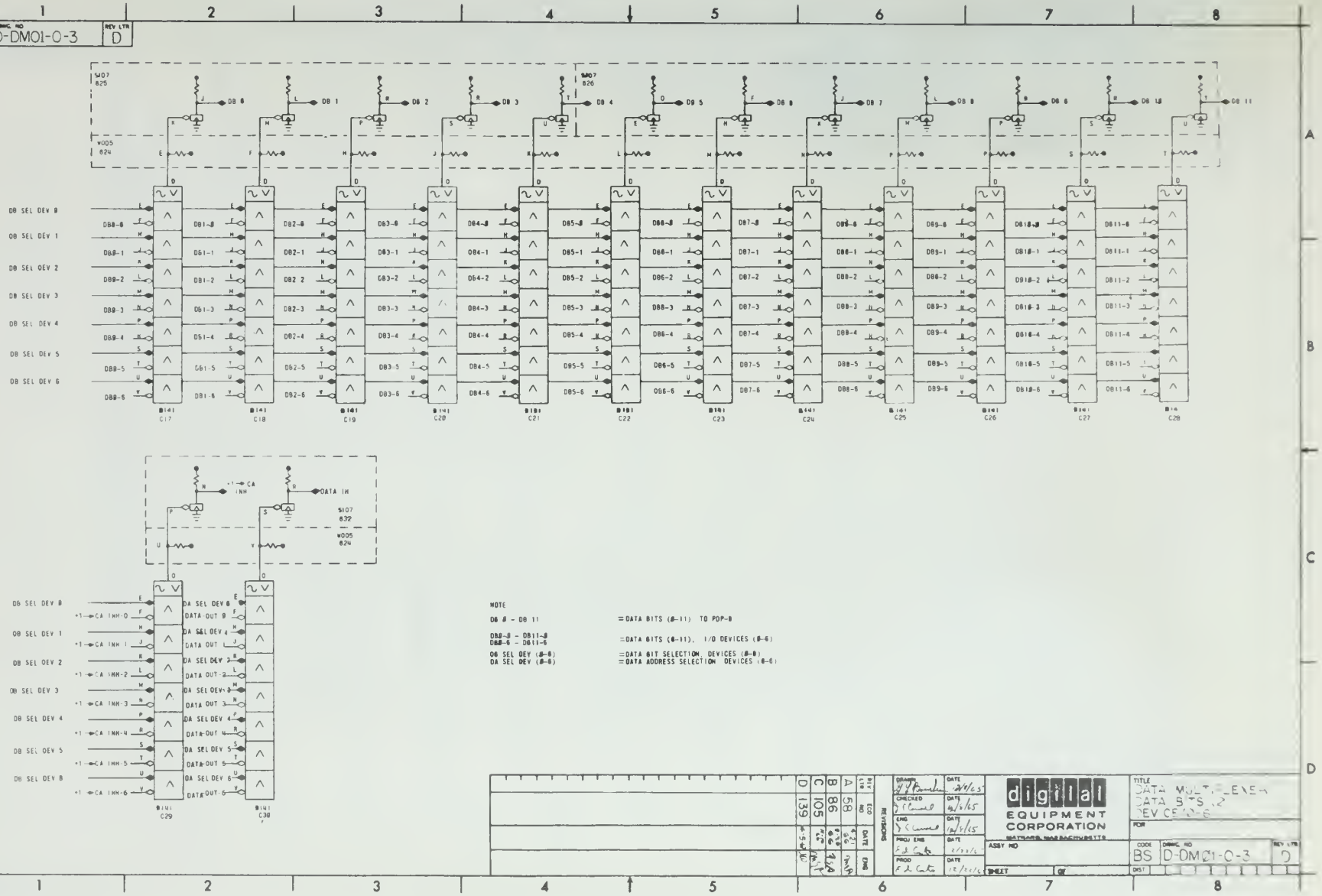
NOTES:

1. WIRING SIDE OF CARD-OUTPUT PINS ARE LETTERS (A = +5V; C = GRD).
2. COMPONENT SIDE OF CARD-OUTPUT PINS ARE NUMBERS (1 = +5V; 2 = GRD).

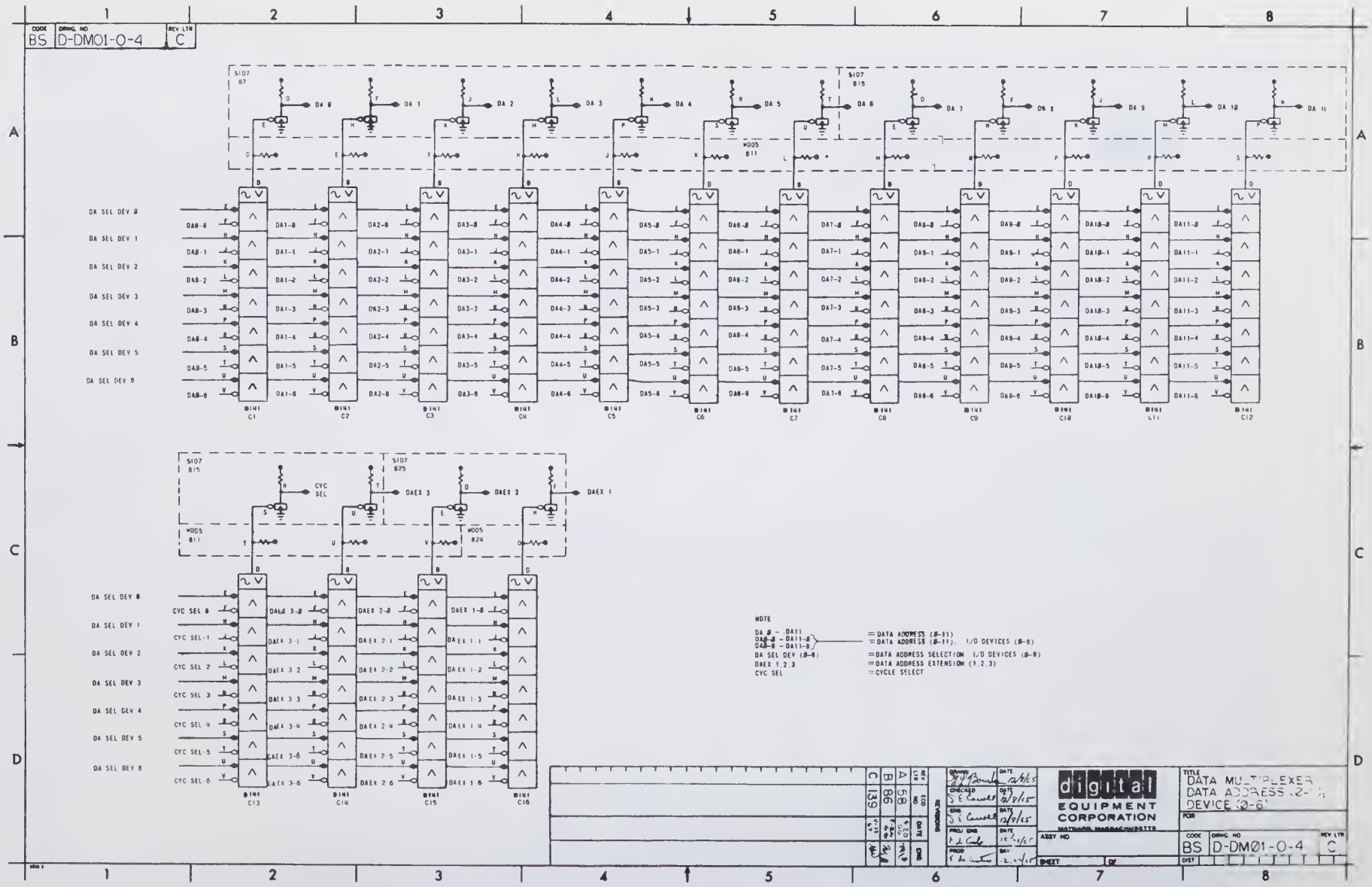
PDP-8 INTERFACE TO 2701
DATA ADDRESS REGISTER



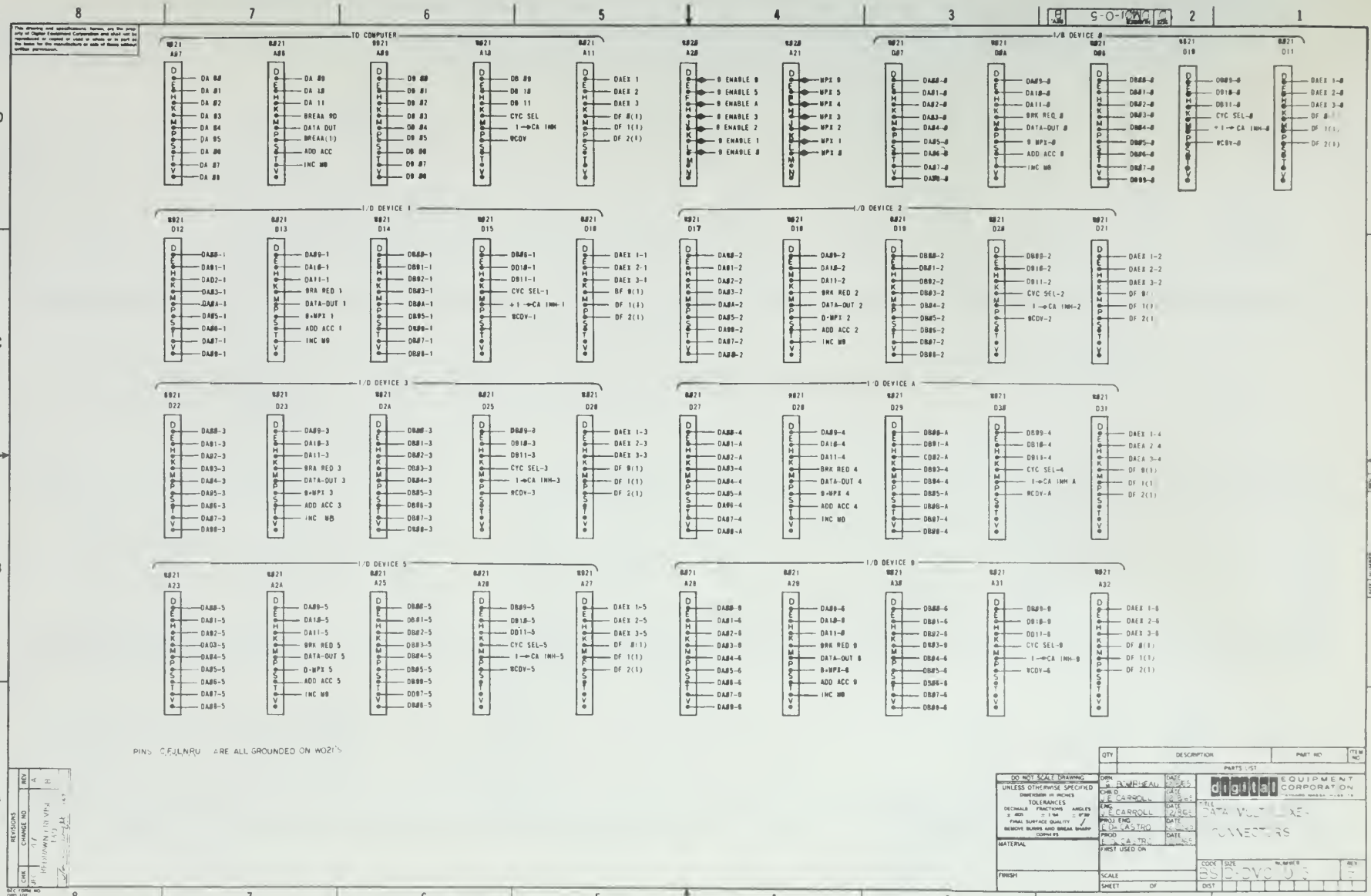
Multiplexer Control (BS-D-DM01-0-2)



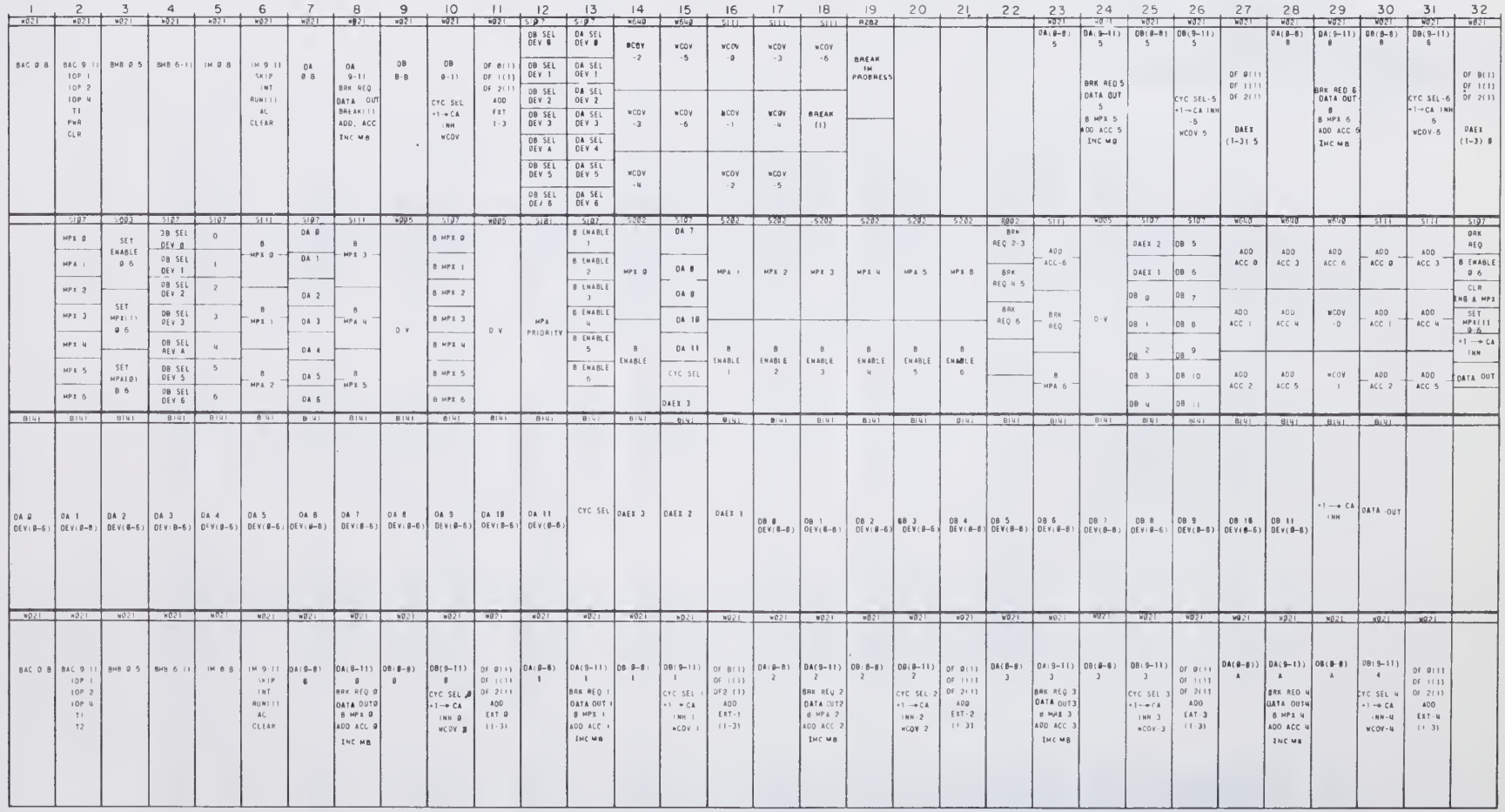
Data Bits Line Selector (BS-D-DM01-0-3)



Data Address Line Selector (BS-D-DM01-0-4)



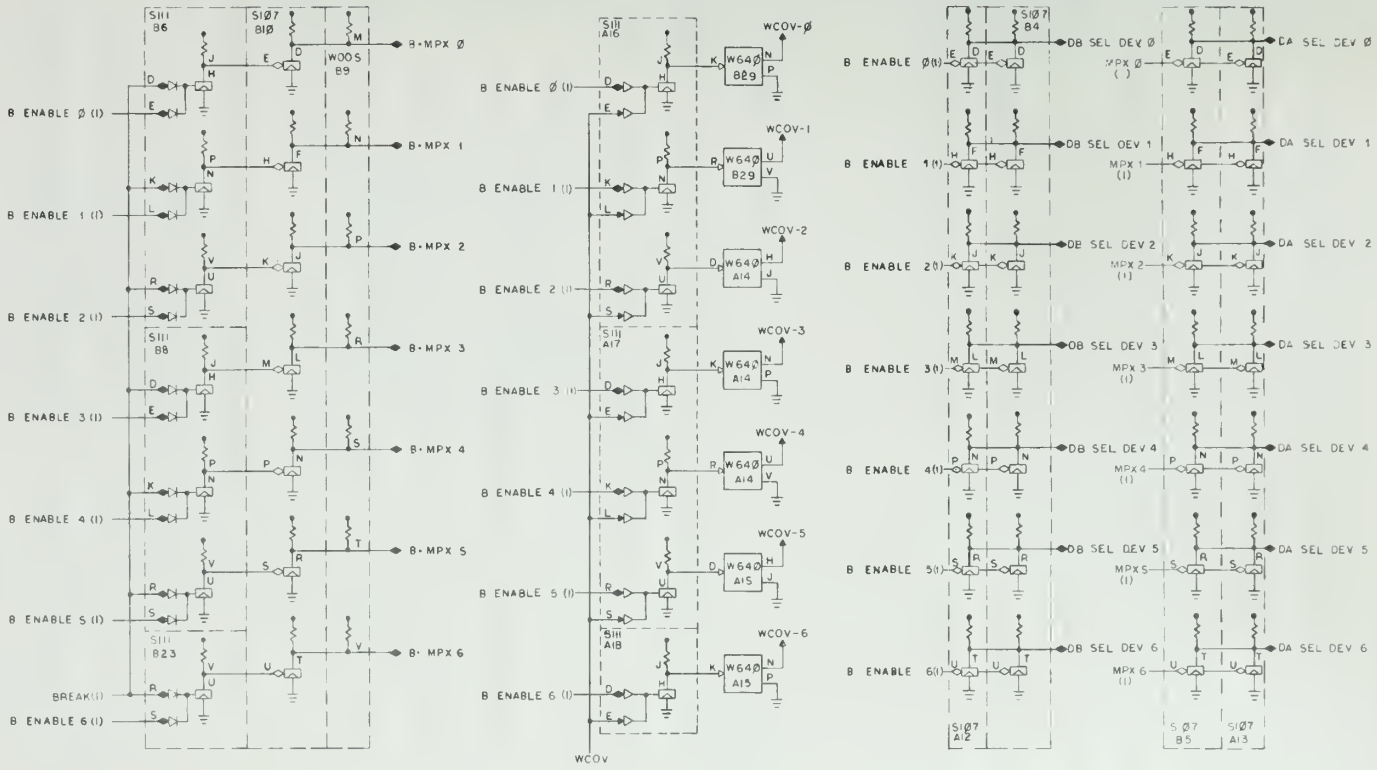
Data Multiplexer Connectors (BS-D-DM01-0-5)



Utilization Module List (UML-D-DM01-0-8)

Level Production from Multiplexer Control (BS-D-DM01-0-11)

CODE BS | DRWG NO D-DM01-0-11 | REV LTR C



NOTE:
DA SEL DEV (Ø-6) = DATA ADDRESS SELECTION I/O DEVICE (Ø-6)
DB SEL DEV (Ø-6) = DATA BIT SELECTION I/O DEVICE (Ø-6)

REVISIONS				
REV LTR	ECO NO.	DATE	ENG
C	139	4-5-65	
B	86	4-2-65	
A	58	4-2-65	

DRAWN B.LUMPKIN DATE 9-14-65
CHECKED DATE
ENG DATE
PROJ ENG DATE
PROD DATE

digital EQUIPMENT CORPORATION MAYNARD, MASSACHUSETTS

ASSY NO
SHEET OF

TITLE DATA MULTIPLEXER TYPE DM01
FOR
CODE BS | DRWG NO D-DM01-0-11 | REV LTR C
DIST

2. PROGRAMMING OF CHANNEL

2.1 General Description

The PDP-8 is equipped with a communications channel to the 360. Since it is essentially half-duplex, data may be carried on this channel in only one direction at a time. To assist in the programming of this channel, it has been constructed to appear at the PDP-8 end as a full-duplex channel when all instructions are used as recommended. From the 360 end, the channel appears as half-duplex. For this reason the 360 must be careful not to attempt to perform more than one input/output operation at a time to the channel.

2.2 Data Transfer

From 1 to 4096 PDP-8 words can be transferred in a single channel I/O operation to or from any address in PDP-8 memory. For transfers to the PDP-8 ("channel write"), each pair of 360 bytes (16 bits) represents one PDP-8 word; conversely, for transfers to the 360 ("channel read"), each PDP-8 word is packed into two 360 bytes (c.f. Figure 5). Essentially, the high order two bits of each 360 byte are unused (ignored on write; set to zero on read).

After an I/O operation is begun, data is moved from the 360 to the PDP-8 (e.g. for a "write") one word (two 360 bytes) at a time. The average maximum data rate is about 12 μsec per word, assuming a 360 multiplexor channel is used, or about 5 μsec for a selector channel (c.f. Figure 6).

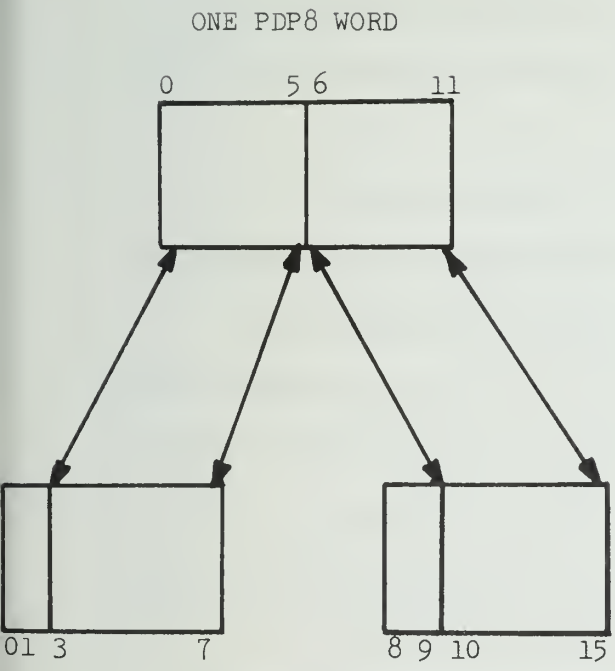


Figure 5

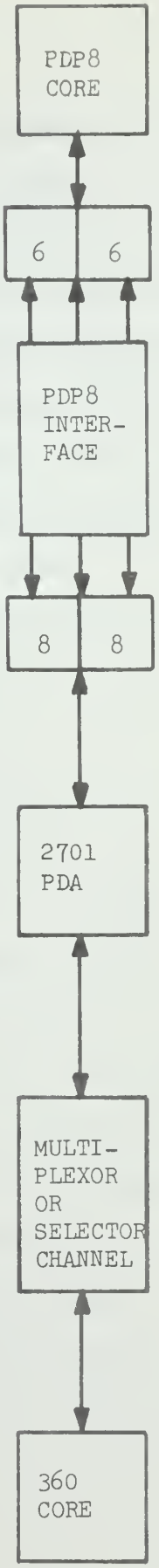


Figure 6

2.3 Communication

2.3.1 Basic Aspects of Communication

The 360 has four channel commands that may be used with the 2701: Read, Read-with-Timeout, Write, and Write-with-Timeout. The general philosophy for the channel states that when the channel is inactive, it is always feasible to issue a Write command to the channel. It is also feasible to issue a Read command at any time that is desired, although it is more common to wait until the PDP-8 has something to say.

When the 360 begins a transfer (issues an SIO instruction) the following sequential events occur:

1. The channel fetches, from the channel program for the transfer, the 360 address of the data; the byte length of the data; and the operation (read, write, etc.).
2. The channel informs the 2701 of its intent to perform the actual transfer. (Note: the byte length and data address are not sent to the 2701).
3. The 2701 informs the PDP-8 interface that the 360 is ready for a transfer.
4. The PDP-8 interface sets read- or write-ready status in the PDP-8.
5. The PDP-8 (software) executes IOT's to give the PDP-8 interface:
   a. a 15-bit address
   b. a 12-bit word count
   c. a "start transfer" command.

6. The PDP-8 interface sends "ready" to the 2701 which sends "ready" to the channel (write) or
   The PDP-8 interface sends a word to the 2701 which sends it to the channel (read).
7. The channel sends a word (actually two bytes, one after another) to the 2701 (write) or the channel accepts the input word (one byte at a time) from the 2701.
8. If the channel's byte count is not yet zero, proceed to Step 3 and continue (but omit Step 5) or
   If the channel's byte count is zero, send "end" to the 2701, which sends "end" to the PDP-8 interface, which sets read- or write-done in the PDP-8.
9. The 360 and the PDP-8 test channel status for successful transfer.

NOTES:

a. After both the 360 and the PDP-8 are aware that a transfer is desired (Steps 1-4), it is the PDP-8's responsibility (and <u>option</u>) to start the actual transmission of data. If the PDP-8 ignored the 360, the channel would wait (possibly forever) until the 360 issued a Halt I/O (HIO) command or until the channel timed out (if read- or write-with-timeout was used). Timeout for the 2701 is about two seconds.

b. Although the PDP-8 interface is "interrupted" for <u>each word</u> in a transfer, the PDP-8 itself (i.e. the software) only investigates the channel's status at the beginning and at the end ("done") of the transfer.

c. The PDP-8 interface and the 2701 obviously transfer only an even number of bytes. The PDP-8 can request only an even number of bytes (since it makes requests in terms of PDP-8 words). However, the 360 has the capability to request the channel to transmit an odd number of bytes. In this case, the channel would read or write only the exact number of bytes requested in the 360 memory. On read, the channel would ignore the last half of the last word presented by the 2701; on write, the last half of the last word presented by the 2701 to the PDP-8 interface is packed with zeros.

d. The 360 has the capability of informing the PDP-8 of its intention to perform a transfer in either direction. The PDP-8, however, can only inform the 360 about its intention to send a message (channel read). See sections 2.3.2.1 (ERI) and 2.3.3. The 360 normally is programmed to initiate a read after it receives this indication from the PDP-8.

2.3.2 PDP-8 Programming

2.3.2.1 Instructions

For channel read:

ERI 6412 Enable Read-Ready Interrupt

Sets read-ready in the 2701, thereby interrupting the 360 indicating the PDP-8's intention of sending a message (i.e. 360 "Attention"). After

execution of this IOT, the 360 would normally begin a channel read operation as outlined in Section 2.3.1. The IOT also enables interrupt on "read select" (i.e. when the PDP-8 interface informs the PDP-8 that the 360 is ready to initiate a read (Step 4 in 2.3.1)), and enables the skip-on-read-ready IOT (SRR).

SRR 6411 Skip on Read Ready

Causes the PDP-8 to skip the next sequential instruction if "Read-Select" has been received (see "ERI"). However, the skip will only occur if an ERI is in effect.

DRI 6422 Disable Read Ready Interrupt

Prevents "Read Select" from causing a PDP-8 interrupt and disables the SRR IOT. Essentially allows the PDP-8 to ignore 360 Read requests (the initial one and those during a transfer). It has nothing to do with "Attention" (e.g. does not undo a previous ERI with respect to "Attention").

ICR 6403 Initiate Channel Read

Start data transfer. The PDP-8 accumulator contains the low order 12 bits of the data address (see SEM IOT below). The accumulator

is cleared after the IOT is performed. This IOT would be issued after "Read Select" has been received (i.e. an ERI was issued; then the SRR skipped). A DRI should be issued before the ICR, so that the PDP-8 will not be interrupted and the SRR will not skip during the transmission of the data.

SRD 6421 Skip on Read Done

Causes the PDP-8 to skip the next sequential instruction if "Read Done" has been received (a read transfer begun by an ICR has completed). "Read Done" will interrupt the PDP-8 if the interrupt is on (ION).

CRD 6414 Clear Read Done

Clears "Read Done" condition. After this IOT, SRD will not skip, and the current "Read Done" will not interrupt the PDP-8.

For Channel Read or Write:

SEM 6434 Set Extended Memory

Sets the high order 3 bits (bank bits) of the data address from bits 6-8 of the PDP-8 accumulator. The accumulator is cleared after the IOT is performed.

SPE 6431 Skip on Parity Error

Causes the PDP-8 to skip the next sequential instruction if the PDP-8 interface or the 2701 detected an error on the previous operation (whether read or write). This IOT is issued after SRD or SWD (see below) skips, thus indicating the completion of an operation. If an error did occur, the operation should be retried. Note: both CPU's get an indication that an error occurred, so both software packages know whether or not a retry will be performed. The occurrence of a transmission error does not interrupt the PDP-8.

CPE 6432 Clear Parity Error

Clears the "Error" condition, so SPE will not skip.

For Channel Write;

EWI 6512 Enable Write-Ready Interrupt

Allows the PDP-8 interface to accept "Write Ready" from the 2701. If the 360 had "recently" or does subsequently initiate a channel write, the PDP-8 would be interrupted with a "Write Select" condition. This IOT also enables the skip-on-write-ready IOT (SWR). Unlike the ERI, no indication is sent to the 360 itself (i.e. no action as with "Attention").

SWR	6511	Analog of SRR
DWI	6522	Analog of DRI
ICW	6503	Analog of ICR
SWD	6521	Analog of SRD
CWD	6514	Analog of CRD

2.3.2.2 Programming Example--Interrupt Feature not Employed

Figure 7 shows an example of a program that sends and receives information from the 360 via a buffer located at octal location 7000, bank 2. The ERI instruction is not included for its affect on interrupt, but for its use in signaling the 360. The EWI instruction in the second program segment is used to enable the SWR.

2.3.2.3 Programming with Interrupt

The program in Figure 8 accomplishes the same function as that in Figure 7, except that it is not necessary to wait until the transfers are completed. The interrupt processor will initiate the desired transfer when the 360 starts one of the input/output operations. Since the 360 can start only one such transfer at a time, both transfers may be requested simultaneously.

Please note that this program does not make use of program flags and switches which are essential to determine whether a transfer is in progress before issuing some sort of initiating sequence such as ERI and EWI. Also, it is possible that the 360 might not fully cooperate, so some checks should be made before going directly into an ICR or ICW after the appropriate skip instruction.

```
/PERFORM CHANNEL "READ".
GO,             ERI             /SIGNAL DESIRE TØ SEND
                SRR             /SKIP IF 360 READY
                JMP     .-1     /IF NØT TEST AGAIN
                DRI             /DISABLE INTERRUPT
                TAD     BNK     /BANK BITS
                SEM             /SET BANK BITS
                TAD     BUF     /ADDRESS ØF DATA
                ICR             /INITIATE TRANSFER
                SRD             /WAIT FØR END
                JMP     .-1
                CRD             /CLEAR DØNE FLAG
                (SPE            /TEST FØR ERRØR)
                  :
BNK,    20
BUF,    7000
/PERFØRM CHANNEL WRITE.
                EWI             /ENABLE THE SWR
                SWR             /WAIT FØR 360 TØ SEND
                JMP     .-1     /LØØP UNTIL READY
                TAD     BNK     /BANK BITS
                SEM             /SET BANK BITS
                TAD     BUF     /ADDRESS ØF BUFFER
                ICW             /INITIATE TRANSFER
                SWD             /LØØP UNTIL CØMPLETE
                JMP     .-1
                CWD             /CLEAR "WRITE DØNE"
                (SPE            /TEST FØR ERRØR
                  :
```

Figure 7

PROGRAM TO SEND TO THE 360 FOLLOWED BY A PROGRAM TO RECEIVE FROM THE 360 WITH PDP8 INTERRUPT OFF

```
INTHND,          .
                 .
                 .
TESTRR,          SRR                 /SEE IF READY TØ SEND
                 JMP       TESTRD
                 DRI                 /IF SØ DISABLE INTERRUPTS
                 TAD       BNK
                 SEM                 /SET BANK BITS
                 TAD       BUF       /GET ADDRESS ØF BUFFER
                 ICR                 /START TRANSFER
                 JMP       RETURN    /GØ BACK TØ PRØGRAM
TESTRD,          SRD                 /SEE IF "READ" FINISHED
                 JMP       TESTWR
                 CRD                 /CLEAR DØNE AND
                 JMP       RETURN    /EXIT
TESTWR,          SWR                 /SEE IF READY TØ RECEIVE
                 JMP       TESTWD
                 DWI                 /DISABLE INTERRUPTS
                 TAD       BNK
                 SEM
                 TAD       BUF       /ADDRESS ØF BUFFER
                 ICW                 /INITIATE TRANSFER
                 JMP       RETURN
TESTWD,          SWD                 /SEE IF "WRITE" IS DØNE
                 JMP       ØTHERS    /GØ TEST ØTHER INTERRUPTS
                 CWD                 /CLEAR "DØNE" AND
                 JMP       RETURN    /EXIT
                 .
                 .
                 .
START,           ERI                 /PREPARE BØTH ØF
                 EWI                 /THE I/Ø ØPERATIØNS
                 .
                 .
                 .
```

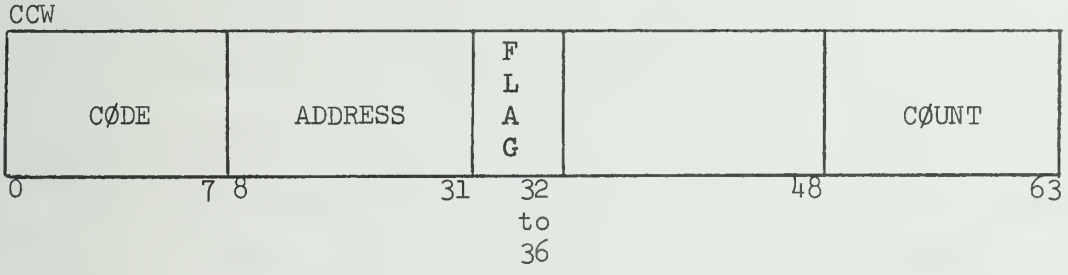
Figure 8

READ FOLLOWED BY WRITE USING INTERRUPT

2.3.3 360 Programming

2.3.3.1 Instructions

Programming of 360 channels and the 2701 is completely standard and is described in the publication IBM System 360 Principles of Operation, Form No. A22-6821. Significance of all bits in the channel status word (CSW), channel address word (CAW), and the channel command word (CCW) is identical to that for any channel. For this reason only a cursory explanation of the chain of events will be given. The command codes which are supplied in the first byte of the CCW are given in Figure 9.



Command Code - Bits 0 to 7 = (in hexadecimal)

1 - Write Command

11 - Write-with-Timeout

2 - Read Command

12 - Read-with-Timeout

Figure 9

COMMAND CODES FOR CHANNEL COMMANDS FOR PDP8 CHANNEL

Since most programming of the 360 is not done in a stand-alone fashion, but is done in connection with the supplied IBM operating system,

and because a special programming problem exists in this connection, a special note on in-system programming will follow later.

The important thing to remember in programming on the 360 is that it is completely different from the PDP-8 in its simplicity. To read, one invokes the READ command. The same applies to write. The only rule to observe is never start a transfer while a transfer is already taking place.

2.3.3.2 Basic Sequence of Operations

In order to initiate a transfer, the 360 executes the SIO instruction. When this occurs, the channel specified by SIO fetches the address of the CCW from the channel address word (CAW) which is at location 72 decimal. The channel then executes the CCW, which contains the address and length for the data buffer in 360 storage. When the transfer terminates and the interrupt for the device occurs, the status for the device will be deposited into the CSW (channel status word) in location 64 decimal. Status contains such flags as channel end, device end, and unit check. Such indications as timeout are obtained through issuance of a sense command (a standard CCW). Timeout for the 2701 is the low order bit of the first sense byte.

2.3.3.3 Programming with IBM Operating System

In order to execute a channel command (CCW), the programmer simply issues an ECP macro-instruction (fully described in IBM System /360 Operating System, System Programmers Guide, Form C28-6550) which

indirectly specifies the CCW to be executed. Upon termination of the operation, the system automatically gives the channel status word and the results of a sense command at this time.

The only requirement that is not easily satisfied is detection of "Attention." The attention condition at one time was closely integrated with the operation of an input/output device being used during individual operations. In the 360, however, an attempt was made to restrict it to functions that do not concern executing programs, such as "request" on an online console typewriter or indications of a disk drive coming into "ready." For this reason, "Attention" has been made very difficult to access, and a modification of the operating system was necessary. It is interesting to note that IBM has produced a channel-to-channel adapter (CTC) which serves much the same purpose as the PDP-8 to 360 channel described herein, and utilizes "Attention" in a very similar fashion. At the time of this writing, however, this device is not supported by IBM software. All programs using the CTC including ASP make use of the software change about to be described.

When "Attention" causes an interrupt, the address of the interrupt handler is taken from the input/output new PSW. The trick is to provide the address of an interception routine and place it in this PSW. This routine should check the I/O old PSW at location 56 decimal to see if the channel and device causing the interruption are the ones of interest. If so, the CSW at location 64 may immediately be tested for "Attention." If it is on, a program flag may be set. When the routine is finished, it should transfer to the location of the IBM I/O interrupt handling the routine originally specified in the I/O new PSW at location 120 decimal.

APPENDIX A
PDP-8 COMPUTER AND DATA MULTIPLEXOR

PDP-8 INTRODUCTION

The Digital Equipment Corporation Programmed Data Processor-8 (PDP-8) is designed for use as a small-scale general-purpose computer. It is a one-address, fixed word length, parallel computer using 12 bit, two's complement arithmetic. Cycle time of the 16K word random-address magnetic-core memory is 1.5 microseconds. Standard features of the system include indirect addressing and facilities for instruction skipping and program interruption as functions of input-output device conditions.

COMPUTER ORGANIZATION

The PDP-8 system is organized into a processor, core memory, and input/output equipment and facilities. All arithmetic, logic, and system control operations of the standard PDP-8 are performed by the processor. Permanent (longer than one instruction time) local information storage and retrieval operations are performed by the core memory. The memory is continuously cycling and automatically performing a read and write operation during each computer cycle. Input and output address and data buffering for the core memory is performed by registers of the processor, and operation of the memory is under control of timing signals produced by the processor.

Interface circuits for the processor allow bussed connections to a variety of peripheral equipment. Each input/output device is responsible for detecting its own select code and for providing any necessary input or output gating. Individually programmed data transfers between the processor and peripheral equipment take place through the processor accumulator. Data transfers can be initiated by peripheral equipment rather than by the program, by means of the

the data break facilities. Standard features of the PDP-8 also allow peripheral equipment to perform certain control functions such as instruction skipping and a transfer of program control initiated by a program interrupt.

MAJOR REGISTERS

To store, retrieve, control, and modify information and to perform the required logical, arithmetic, and data processing operations, the core memory and the processor employ the logic components shown in Figure and described in the following paragraphs.

Accumulator (AC)

Arithmetic and logic operations are performed in this 12-bit register. Under program control the AC can be cleared or complemented, its content can be rotated right or left with the link. The content of the memory buffer register can be added to the content of the AC and the result left in the AC. The content of both of these registers may be combined by the logical operation AND, the result remaining in the AC. The inclusive OR may be performed between the AC and the switch register on the operator console and the result left in the AC.

The accumulator also serves as an input-output register. All programmed information transfers between core memory and an external device pass through the accumulator.

Link (L)

The one-bit register is used to extend the arithmetic facilities of the accumulator. It is used as the carry register for two's complement arithmetic. Overflow into the L from the AC can be checked by the program to greatly simplify and speed up single and multiple precision arithmetic routines. Under program control the link can be cleared and complemented, and it can be rotated as part of the accumulator.

Program Counter (PC)

The program sequence, that is the order in which instructions are performed,

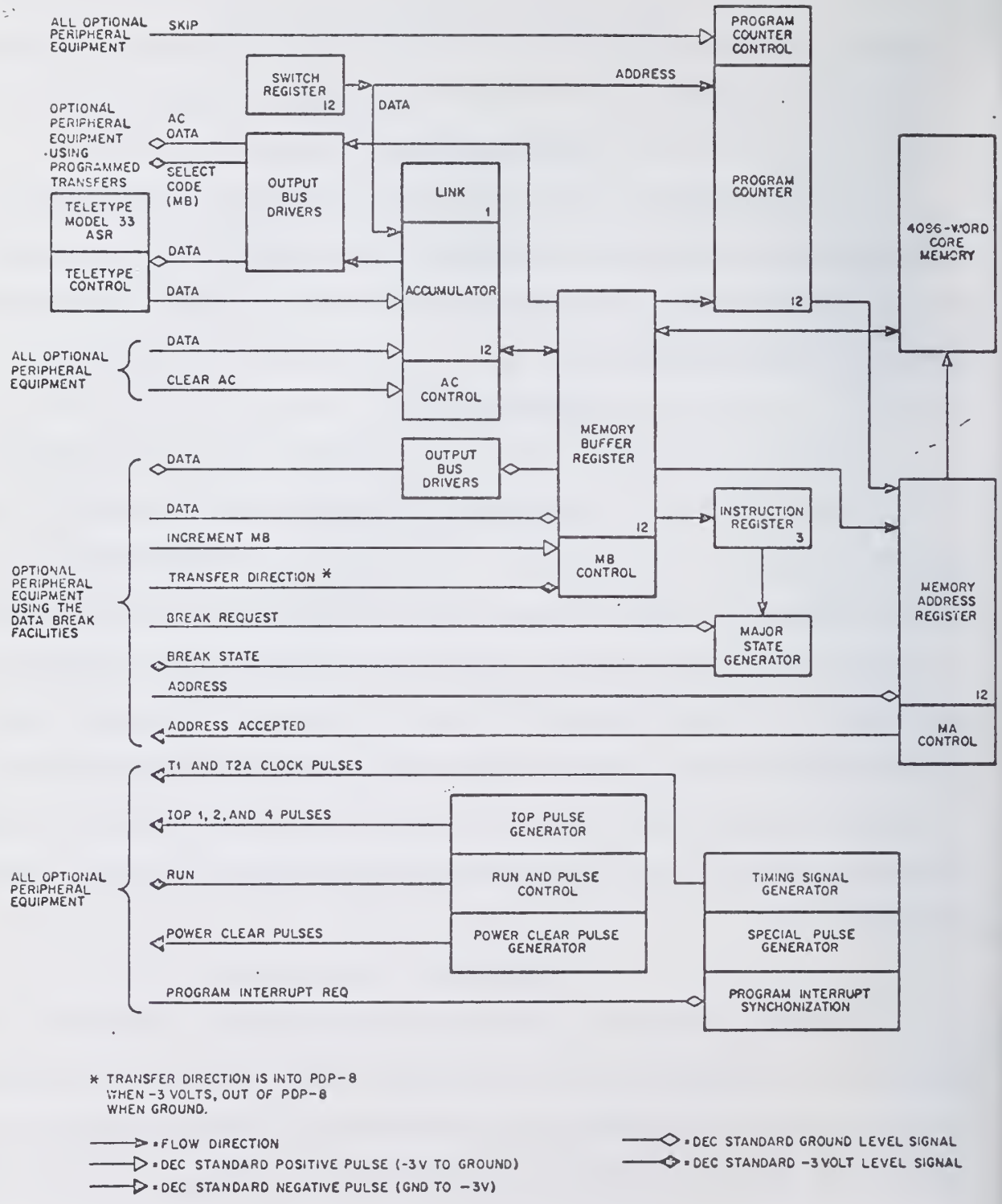


Fig. 10 PDP-8 Block Diagram

is determined by the PC. This 12-bit register contains the address of the core memory location from which the next instruction will be taken. Information enters the PC from the core meory, via the memory-buffer register, and from the switch register on the operator console. Information in the PC is transferred into the memory address register to determine the core memory address from which each instruction is taken. Incrementation of the content of the PC establishes the successive core memory locations of the program and provides skipping of an instruction based upon a programmed test of information or conditions.

Memory Address Register (MA)

The address in core memory which is currently selected for reading or writing is contained in this 12-bit register. Data can be set into it from the memory buffer register, from the program counter, or from an I/O device using the data break facilities.

Switch Register (SR)

Information can be manually set into the switch register for transfer into the PC as an address by means of the LOAD ADDRESS key, or into the AC as data to be stored in core memory by means of the DEPOSIT key.

Core Memory

The core memory provides storage for instructions to be performed and information to be processed or distributed. This random address magnetic core memory holds 16K 12-bit words. Memory location 0 is used to store the content of the PC following a program interrupt, and location 1 is used to store the first instruction to be executed following a program interrupt. (When a program interrupt occurs, the content of the PC is stored in location 0 and program control is transferred to location 1 automatically.) Locations 10_8 through 17_8 are used for auto-indexing. All other locations can be used to store instructions or data.

Core memory contains numerous circuits such as read-write switches, address

decoders, inhibit drivers, and sense amplifiers. These circuits perform the electrical conversions necessary to transfer information into or out of the core array and perform no arithmetic or logic operations upon the data.

Memory Buffer Register (MB)

All information transfers between the processor registers and the core memory are temporarily held in the MB. Information can be transferred into the MB from the accumulator or memory address register. Information can be set into the MB from an external device during a data break or from core memory, via the sense amplifiers. Information is read from a memory location in 0.8 microseco and rewritten in the same location in another 0.8 microsecond of one 1.6 microseco memory cycle.

Instruction Register (IR)

The 3-bit register contains the operation code of the instruction currently being performed by the machine. The three most significant bits of the current instruction are loaded into the IR from the memory buffer register during a Fetch cycle. The content of the IR is decoded to produce the eight basic instructions, and affect the cycles and states entered at each step in the program.

Major State Generator

One or more major control states are entered to determine and execute an instruction. The major state generator determines the machine state during each computer cycle. The major states are Fetch, Defer, Execute, and Break. Each state is produced as a function of the current instruction and the current state, except for the Break state which is entered upon receipt of the Break Request signal supplied by peripheral equipment.

Fetch

During this state an instruction is read into the MB from core memory

at the address specified by the content of the PC. The instruction is restored in core memory and retained in the MB. The operation code of the instruction is transferred into the IR to cause enactment, and the content of the PC is incremented by one.

If a multiple-cycle instruction is fetched, the following major state will be either Defer or Execute. If a one-cycle instruction is fetched, the operations specified are performed during the last part of the Fetch cycle and the the next state will be another Fetch.

Defer

When a 1 is present in bit 3 of a memory reference instruction, the Defer state is entered to obtain the full 12-bit address of the operand from the address in the current page or page 0 specified by bits 4 through 11 of the instruction. The process of address deferring is called indirect addressing because access to the operand is addressed indirectly, or deferred, to another memory location.

Execute

This state is entered for all memory reference instructions except jump. During an AND, two's complement add, or increment and skip if zero instruction the content of the core memory location specified by the address portion of the instruction is read into the MB and the operation specified by bits 0 through 2 of the instruction is performed. During a deposit and clear accumulator instruction the content of the AC is transferred into the MB and is stored in core memory at the address specified in the instruction. During a jump subroutine instruction this state occurs to write the content of the PC into the core memory address designated by the instruction and to transfer this address into the PC to change control.

Break

When this state is established, the sequence of instructions is broken for a data break. The break occurs at the completion of the current instruction.

The data break allows information to be transferred between core memory and an external device, via the MB. When this transfer has been completed, the program sequence is resumed from the point of the break.

Output Bus Drivers

Output signals from the computer processor are power amplified by output bus driver modules of the standard PDP-8; allowing these signals to drive a heavy circuit load.

Functional Summary

Operation of the computer is accomplished on a limited scale by keys on the operator console. Operation in this manner is limited to address and data storage by means of the switch register, core memory data examination, the normal start/stop/continue control, and the single step or single instruction operation that allows a program to be monitored visually as a maintenance operation. Most of these manually initiated operations are performed by executing an instruction in the same manner as by automatic programming, except that the gating is performed by special pulses rather than by the normal clock pulses. In automatic operation, instructions stored in core memory are loaded into the memory buffer register and executed during one or more computer cycles. Each instruction determines the major control states that must be entered for its execution. Each control state lasts for one 1.5-microsecond computer cycle and is divided into distinct time states which can be used to perform sequential logical operations. Performance of any function of the computer is controlled by gating of a specific instruction during a specific major control state and a specific time state.

TIMING AND CONTROL ELEMENTS

Figure shows the timing and control elements described in the succeeding paragraphs and indicates their relationship to the major registers. These elements

can be grouped categorically into timing generators, register controls, and program controls.

Timing Generators

Timing pulses used to determine the computer cycle time and used to initiate sequential time-synchronized gating operations are produced by the timing signal generator. Timing pulses used during operations resulting from the use of the keys and switches on the operator console are produced by the special pulse generator. Pulses that reset registers and control circuits during power turn on and turn off operations are produced by the power clear pulse generator. Several of these pulses are available to peripheral devices using programmed or data break information transfers.

Register Controls

Operation of the AC,MA,MB, and PC is controlled by an associated logic circuit. These circuits, in turn, transmit and receive control signals to and from I/O equipment. Programmed data transfer equipment can supply a pulse to the AC control to clear the AC prior to a data input and can supply a pulse to cause the content of the PC to be incremented, thus initiating an instruction skip. Equipment using the data break facility passes signals with the MA control and MB control to determine the direction and timing of data transfers in this mode.

Program Controls

Circuits are also included in the PDP-8 that produce the I/O pulses which initiate operations involved in input/output transfers, determine the advance of the computer program, and allow peripheral equipment to cause a program interrupt of the main computer program to transfer program control to a subroutine which performs some service for the I/O device.

INTERFACE

The input/output portion of the PDP-8 is very flexible and interfaces

readily with special equipment, especially in real time data processing and contro environment.

The PDP-8 utilizes a "bus" I/O system rather than the more conventional "radial" system. The "bus" system allows a single set of data and control lines to communicate with all I/O devices. The bus simply goes from one device to the next. No additional connections to the computer are required.

External devices receive two types of information from the computer: data and control signals. Computer output data is present as static levels on 12 lines These levels represent a 12-bit word to be transmitted in parallel to a device. Data signals are of two types: levels and timing pulses. Six static levels and their complement are supplied by the MB on 12 lines. These lines contain a code representing the device from which action is required. Each device recognizes its own code and performs its function only when this code is present. There are three timing pulses which may be programmed to occur. These IOP pulses are separated in time by one microsecond and are brought to all devices on 3 lines. These pulses are used by a device only when it is selected by the appropriate code on the level lines. These may be used to perform sequential functions in an external register, such as clear and read, or any other function requiring one, two, or three sequential pulses.

Peripheral devices transmit information to the computer on four types of "busses". These are the information bus, the clear AC bus, the skip bus, and the program interrupt bus. The information bus consists of 12 lines normally held at -3 volts by load resistors within the computer. Whenever one of these lines is brought to ground, a binary 1 will be placed in the corresponding accumulator bit. Each device may use the input bus only when it is selected; and thus, these input lines are time shared among all of the connected devices. The skip bus is electrically identical to the information bus. However, when

it is driven to ground the next sequential instruction will be skipped. It too can be used only by the device currently selected and is effectively time shared. The program interrupt bus may be driven to ground at any time by any device whether currently selected or not. When more than one device is connected to the interrupt bus they should also be connected to the skip bus so the program can identify the device requesting program interruption.

The transmission of device selection levels and timing pulses is completely under program control. A single instruction can select any one of 64 devices and transmit up to three IOP timing pulses. Since the timing pulses are individually programmable, one might be used to strobe data into an external device buffer, another to transmit data to the computer, and the third to test a status flip-flop and drive the skip bus to ground if it is in the enabling state.

Data transfers may also be made directly with core memory at a high speed using the data break facility. This is a completely separate I/O system from the one described previously. It is standard equipment in every PDP-8 and is ordinarily used with fast I/O devices such as magnetic drums or tapes. Transfers through the data break facility are interlaced with the program in progress. They are initiated by a request from the peripheral device and not by programmed instruction. Thus, the device may exchange a word with memory whenever it is ready and does not have to wait for the program to issue an instruction. Computation may proceed on an interlaced basis with these transfers.

APPENDIX B

2701 Data Adapter Unit

Introduction

A very short description of the 2701 is given in order to indicate the relative position of the Parallel Data Adapter (PDA) to the IBM 360 processor. Most of the material is specifically about the PDA; its functional analysis; its functional sections; its operation.

Functional Sections of the 2701

The three function sections of the 2701 are:

Channel Interface (CHIF)

Transmission Interface Converter (XIC)

Transmission Adapter (XA)

The XIC and XA operate as a couple, which in conjunction with the CHIF, provides a single complete path for the operation of the terminal devices with the I/O channel. A minimum 2701 configuration contains one CHIF and one XIC-XA couple.

Channel Interface (CHIF)---An I/O channel (either a multiplexer channel or a selector channel) is a facility that serves as a means of communication between the System/360 processor/main storage and one or more input/output (I/O) devices. It provides for and controls the interchange of data, control, and program information between the processor/main storage and the I/O devices.

The channel-interface section of the 2701 provides the circuits to attach the 2701 to a System/360 I/O channel. It supplies the path for transferring the various control signals, addresses, commands, and data between the I/O channel and an XIC and also controls the

operation of the usage meter. The CHIF is capable of operating with up to four XIC's and will interface normally with one I/O channel, or two in the event the 2701 is equipped with the Second Channel Interface feature.

Transmission Interface Converter (XIC)---The transmission-interface-converter section of the 2701 controls information and/or controls signal transfers between the System/360 I/O channel (via the CHIF) and a transmission adapter. The XIC operates through the CHIF with either a selector or multiplexer channel. When the XIC is connected to a selector channel, information transfer is always in byte mode; when connected to a multiplexer channel, information transfer is normally in the data-interleave (multiplex) mode. However, in the latter case, an XA can force multiple-byte mode for any number of bytes.

The XIC stores channel commands for the transmission adapter and handles byte transfer to or from main storage when requested by the XA. The XIC also responds to specific commands received from the I/O channel and/or specific requests from the XA, initiates operation-ending procedures when requested by the XA, relays an Interface Stop signal from the I/O channel to the XA, and stores the status byte and a sense byte for transfer to main storage.

Transmission Adapter (XA)---The transmission-adapter section of the 2701 contains circuits necessary for the connection of a remote terminal (station, remote processor, device) to the 2701 and the necessary controls to effect movement of data to or from the channel via the CHIF and XIC. The XA decodes the I/O channel commands presented by the XIC, initiates service requests for data-byte transfer, and provides buffering for each transmitted or received character. Terminal-control functions

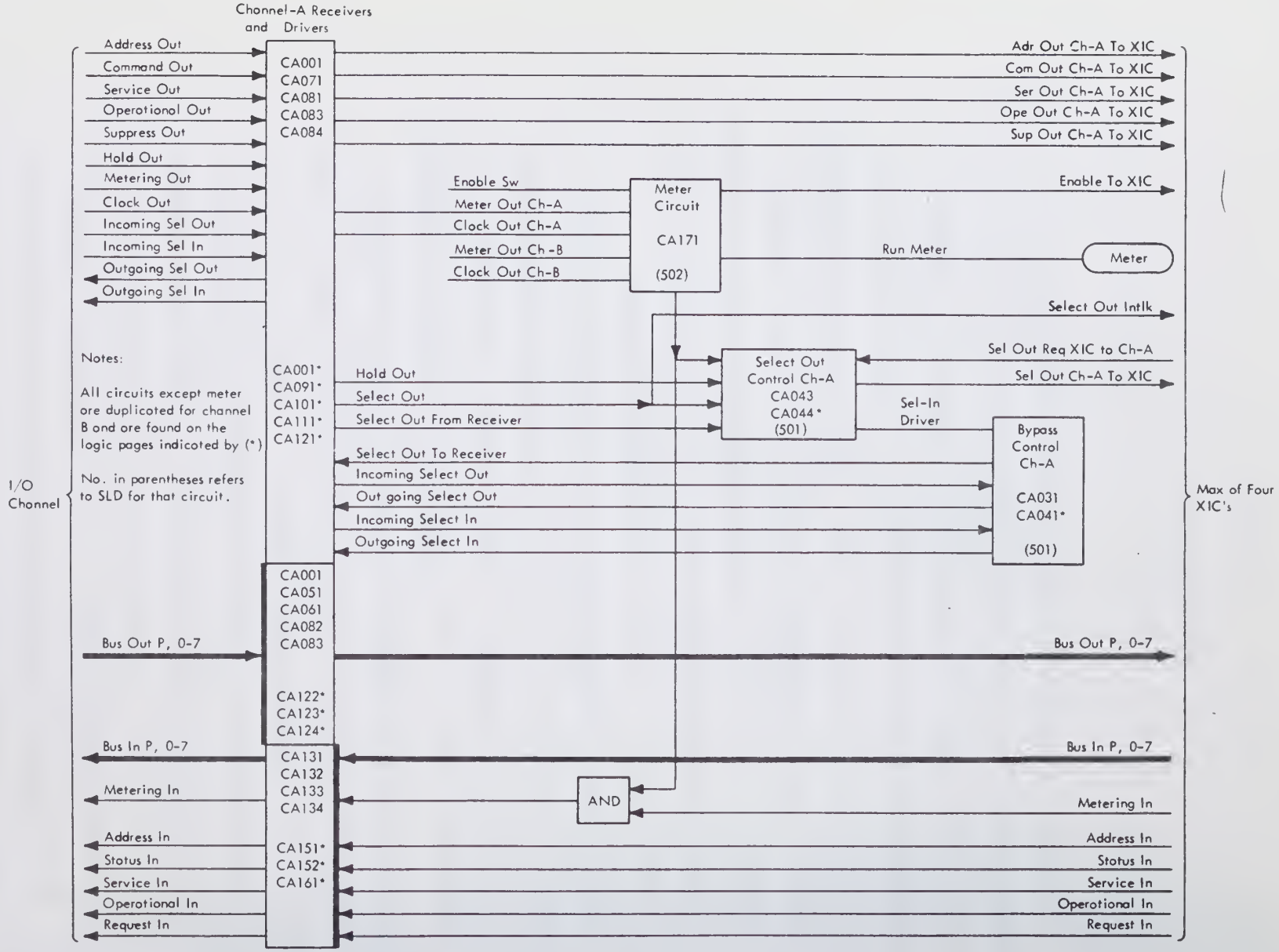


Figure 11 Channel Interface Feature (CHIF)

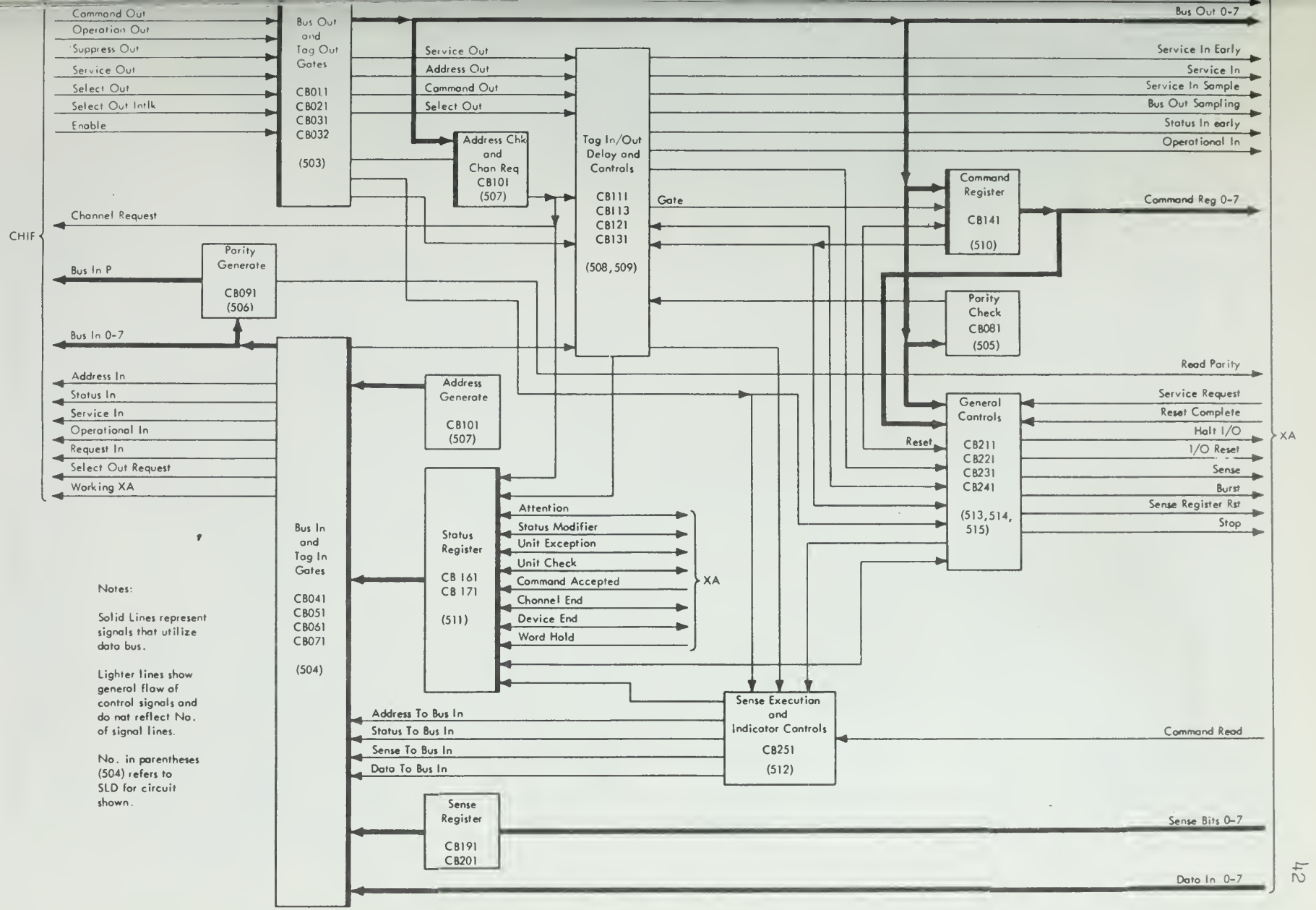


Figure 12 Transmission Interface Control (XIC)

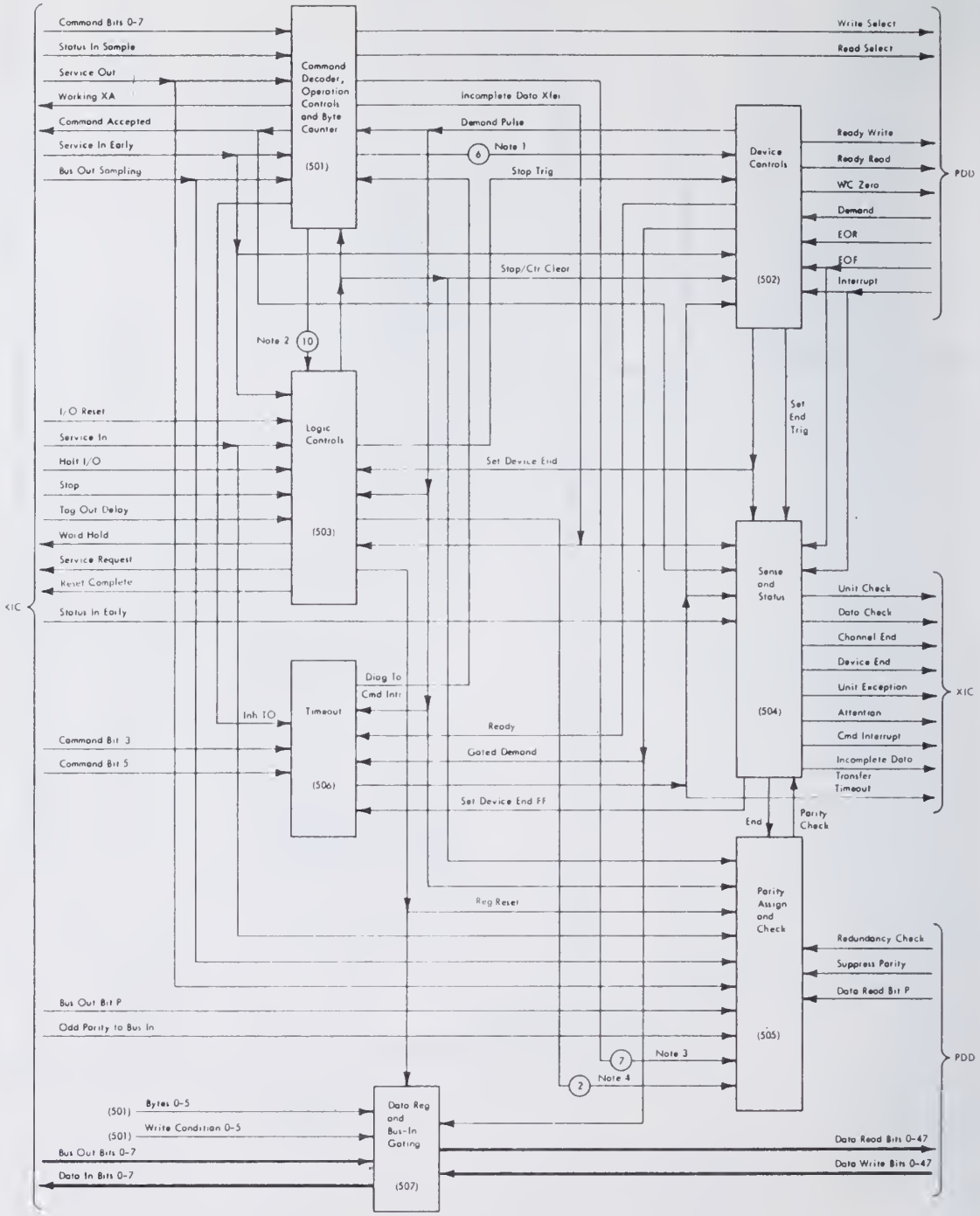


FIGURE 13 UNIT DATA AND CONTROL DIAGRAM

such as communication-interface control, character and character-sequence recognition, parity checking, sense and status byte generation, end-of-operation, and error detection are performed by the XA.

Parallel Data Adapter

Introduction---The Parallel Data Adapter operates with and is partially controlled by the Transmission Interface Converter (XIC) section of the 2701 Data Adapter Unit.

The Parallel Data Adapter presents the external device with a demand/response interface to allow for a half-duplexed exchange of data between the device and the System/360 processor.

The IBM 2701 Parallel Data Adapter (PDA) is a Transmission Adapter (XA) of the Data Acquisition and Control type. The PDA serves as a high-speed, variable-word-length buffer between the XIC and the Parallel Data Device (PDD). Data from the processor is received from the XIC in 8-bit bytes, is formed into 16-bit words, and is transferred to the PDD parallel-by-bit and serial-by-word over a demand/response type interface. Conversely, data from the PDD is broken into 8-bit bytes and is transferred to the System/360 CPU via the XIC.

The PDA interprets the I/O channel command presented by the XIC, initiates a Request Service for data byte transfers, performs the necessary operations to form the bytes into a word (or, during execution of a Read command, to break a word into bytes), then transfers the data to the PDD or to the CPU. When an end-of-operation sequence is initiated by the XIC or the PDD, sense and/or status bits are set to indicate that the sequence has occurred.

Functional Analysis of 2701 PDA Operations---The XIC initiates any of the operations that the PDA is capable of performing by placing

a valid command on the Command Register lines. The reception and decoding of any valid command causes the PDA to generate a Command Accepted level and to return it to the XIC. If other than a valid command is placed on the Command Register lines, it is rejected, i.e., the Command Accepted signal is not produced and the XIC sends Command Rejected to the channel. The commands that a PDA can decode and execute are listed in Table 1 and are described below.

TABLE 1 COMMAND BIT CONFIGURATIONS

Command	Bits 0	1	2	3	4	5	6	7
Read	0	0	0	0	0	0	0	0
Read with Timeout*	0	0	0	1	0	0	1	0
Write	0	0	0	0	0	0	0	1
Write With Timeout*	0	0	0	1	0	0	0	1
Diagnostic Read	0	0	0	0	0	1	1	0
Diagnostic Write	0	0	0	0	0	1	0	1

*Not recognized by the PDA if Timeout feature is not installed.

(Read Command). This command controls circuit operations which allow data transfers from the PDD to the PDA and, subsequently, to the XIC and System/360 CPU.

(Read with Timeout Command). This command operates the same as a Read command; however, if the PDD does not respond with data within 2 seconds, the Timeout sense bit is activated and the operation is terminated.

(Write Command). This command controls circuit operations which allow data transfers from the XIC to the PDA and, subsequently, to the PDD.

(Write With Timeout Command). This command operates the same as the Write command; however, if the PDD does not accept the data within 2 seconds, the Timeout sense bit is set and the operation is terminated.

(Diagnostic Read Command). The operation of this command is similar to that of a Read operation; the chief difference is that data transfer and control signals between the PDA and the PDD are inhibited. The command should be given following a Diagnostic Write Command. This causes the contents of the data register to be transferred to the XIC, thereby enabling a thorough check of data transfer from the XIC to the PDA and from the PDA to the XIC.

(Diagnostic Write Command). The operation of this command is similar to that of a Write operation; the chief difference is that data transfer and control signals from the PDA to the PDD are inhibited. This command causes a series of data bytes to be supplied by the XIC. When the data word has been assembled in the data register, the PDA sets Device End and Channel End. This command is chained to a Diagnostic Read command so that the CPU can retrieve and examine the data transferred.

Description of Functional Sections---The PDA comprises seven functional sections:

1. Command decoder and byte counter
2. Device controls
3. Logic controls
4. Sense and status register

5. Parity check and assignment

6. Timeout

7. Data register and bus-in gating

(Command Decoder, Operation Controls, and Byte Counter). This section decodes the commands presented by the System/360 CPU via the XIC, initiates execution of the decodes command, and controls the byte counter and transfer of data from the XIC (Bus Out lines) to the data register. Two of the PDA-PDD interface signals (Write Select and Read Select) are also controlled by this section.

The byte counter is used to determine when the correct number of bytes has been received or transmitted. Jumper wires allow the byte counter to be extended from two to eight counts to accomodate the extension features.

(Device Controls). This section controls the PDA-PDD interface signals (with the exception of those mentioned above and those pertaining to the Parity bit). Control signals are exchanged with other sections of the unit to synchronize PDA operation with PDD operation.

(Logic Controls). This section primarily utilizes signals from the command register and from the device controls to generate Service Request, Word Hold, and Stop signals and to reset the data register.

(Sense and Status Register). This section contains flip-flops which store the Attention (bit 0), Channel End (bit 4), Device End (bit 5), Unit Check (bit 6), and Unit Exception (bit 7) status bits. The status information is transferred to the XIC when the Status In Early interface signal is down. The Command Interrupt (Intervention Required) sense bit (1) and the Data Check sense bit (4) are also generated by this section. The Timeout sense bit (7) and the Incomplete

Data Transfer sense bit (6) are produced by the timeout and command decoder circuits respectively.

(Parity Check and Assignment). This section ascertains the validity of all data transfers between the PDD and XIC. During Write Operations, a parity bit is supplied with every byte transferred to the PDA from the XIC. The Parity circuits sample the status of the XIC Bus Out P line during the transfer of each byte from the XIC to the PDA data register. The parity of the word contained in the PDA data register is presented to the PDD. During Read operations, the PDA sends a data byte to the XIC, where a parity assignment is made for that byte. The parity of that byte is returned to the PDA, and byte transfers continue until the complete word has been transferred to the XIC. A check is then performed by the parity circuits to ascertain that the parity of the word transferred was correct.

(Timeout). The 2-second timeout produced by this section provides time for the PDD to accept or supply data after a Write or Read command is decoded. If the PDD does not respond within this 2-second limit, device malfunction is assumed, and the Timeout signal is sent to the XIC. Special Read and Write commands are used when timeout is required and are operational only on those PDA's on which the timeout feature is installed.

(Data Register). The data register provides a buffer between the XIC and PDD. Eight-bit bytes are received from the XIC and are assembled into a 16-, 24-, 32-, 40-, or 48-bit word (a basic PDA has a 16-bit register; however, additional byte capacity can be ordered). The data register then transfers the word to the PDD under control

of the command decoder section and the byte counter. Data received from the PDD is broken into a series of 8-bit bytes, which are sent to the XIC one byte at a time under control of the device control section

Status Indications---The status bits inform the XIC and CPU when an operation is completed or when an abnormal condition of which the CPU should be aware has occurred. The five status bits utilized by the PDA are Attention, Unit Check, Device End, Channel End, and Unit Exception.

Sense Indications---The sense bits inform the CPU when abnormal or unusual conditions occur within the PDA or the device and further define the indications presented by the status bits.

APPENDIX C

Cost of Interface

Because of the decisions made at the beginning and because of our having done much other interfacing to this same PDP-8 data break multiplexor some savings in hardware cost was realized. Quite a bit of general preparation had been made (prior to this interface) involving shared lines to the multiplexor and accumulator. Some of this cost could be subtracted from the cost of this hardware. The decision to go to integrated circuits for the bulk of the logic saved money as compared to DEC cards.

The cost is divided into four areas:

1.	Integrated Circuits	$252.00
2.	DEC Logic	$230.00
3.	Shared	$ 55.00
4.	Other (racks, cabinets, raw materials) estimate	$500.00
		$1037.00

Starting with a basic PDP-8 might cause this figure to rise to $1260.

1.

Integrated Circuit / Cards	7400	7401	7410	7420	7430	7440	7474	TOTAL
7-13	17				7			24
8-16						4		4
12-4	7			2		1	6	16
12-6	1					8		9
12-7	4					6		10
12-8	3					6		9
12-13						1		1
12-14						6		6
12-15	15		8			1		24
12-16	6		2	1		3	7	19
12-17	5	2	2	1	1	1	3	15
12-18	11					1	1	13
12-20	12				2	1	7	22
	81	2	12	4	10	39	24	172

Based on 100 lots cost = $252

2.

5 - W512 @ 25.00 ea. =	$125.00
1 - R-123 @ 19.00 ea. =	19.00
10 - W-021 @ 4.00 ea. =	40.00
1 - W603 @ 23.00 ea. =	23.00
43 - Hudson two pin bulbs @ .50 ea. =	21.50
Total	$228.50

3.

27 - 7440 @ $1.45 ea.	= $40	÷ 5 =	$ 8.00
4 - W603 @ $23.00 ea.	= $92	÷ 5 =	18.00
4 - R107 @ $24.00 ea.	= $96	÷ 5 =	19.--
12 - W021 @ $4.00 ea.	= $48.	÷ 5 =	10.00
	$276		$55.00

The $55.00 figure is our estimate of cost to us. The $276.00 would be the cost if one were required to start from a basic PDP-8 with no previous I/Ø work done on it.

4.
Other

Because this is an estimate only, some items will be named but no cost will be given.

Printed Circuit Cards
Integrated Circuit Sockets
Wire-wrap Tools
Wire
Racks
Printed Circuit Connectors
IBM Compatable Connectors

Form AEC–427
(6/68)
AECM 3201

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY–TYPE CONTRACTOR'S RECOMMENDATION FOR DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.
COO-1469-0153

2. TITLE
DEC PDP-8 INTERFACE TO IBM 2701 PDA

3. TYPE OF DOCUMENT (Check one):

- [X] a. Scientific and technical report
- [ ] b. Conference paper not to be published in a journal:
  - Title of conference _____
  - Date of conference _____
  - Exact location of conference _____
  - Sponsoring organization _____
- [ ] c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- [X] a. AEC's normal announcement and distribution procedures may be followed.
- [ ] b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
- [ ] c. Make no announcement or distrubution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

C. W. Gear
Principle Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature Charles W Gear.

Date March 1970

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- [ ] a. AEC patent clearance has been granted by responsible AEC patent group.
- [ ] b. Report has been sent to responsible AEC patent group for clearance.
- [ ] c. Patent clearance not required.